



*National Semiconductor*

400053

National  
Ethernet  
Databook





# **ETHERNET DATABOOK**

---

1996

## **ETHERNET PRODUCTS**

**Integrated Network Interface Controller  
Products**

**10 Mb/s Physical Layer Transceivers and  
ENDECs**

**10 Mb/s Repeater Interface Controller  
Products**

**100 Mb/s Fast Ethernet Protocol Products**

## **SUPPORT MATERIALS**

**Glossary and Acronyms**

**Appendix/Physical Dimensions**

**1**

**2**

**3**

**4**

**5**

**6**

## TRADEMARKS

Following is the most current list of National Semiconductor Corporation's trademarks and registered trademarks.

ABIC <sup>TM</sup>	ELSTAR <sup>TM</sup>	MOLE <sup>TM</sup>	SERIES/800 <sup>TM</sup>
Abuseable <sup>TM</sup>	Embedded System Processor <sup>TM</sup>	MPA <sup>TM</sup>	Series 32000 <sup>®</sup>
AirShare <sup>TM</sup>	EPT <sup>TM</sup>	MST <sup>TM</sup>	SIMPLE SWITCHER <sup>®</sup>
Anadig <sup>TM</sup>	E-Z-LINK <sup>TM</sup>	Naked-8 <sup>TM</sup>	SNIT <sup>TM</sup>
APPST <sup>TM</sup>	FACT <sup>TM</sup>	National <sup>®</sup>	SNIC <sup>TM</sup>
ARI <sup>1</sup> <sup>TM</sup>	FACT Quiet Series <sup>TM</sup>	National Semiconductor <sup>®</sup>	SofChek <sup>TM</sup>
ASPECT <sup>TM</sup>	FAIRCAD <sup>TM</sup>	National Semiconductor Corp. <sup>®</sup>	SONIC <sup>TM</sup>
AT/LANTIC <sup>TM</sup>	Fairtech <sup>TM</sup>	NAX 800 <sup>TM</sup>	SpeechPro <sup>TM</sup>
Auto-Chem Deflasher <sup>TM</sup>	FAST <sup>®</sup>	NeuFuz <sup>TM</sup>	SPiKe <sup>TM</sup>
BCPT <sup>TM</sup>	FastLock <sup>TM</sup>	Nitride Plus <sup>TM</sup>	SPiRE <sup>TM</sup>
BI-FET <sup>TM</sup>	FAST <sup>r</sup> <sup>TM</sup>	Nitride Plus Oxide <sup>TM</sup>	Staggered Refresh <sup>TM</sup>
BI-FET II <sup>TM</sup>	GENIX <sup>TM</sup>	NML <sup>TM</sup>	STAR <sup>TM</sup>
BI-LINE <sup>TM</sup>	GNX <sup>TM</sup>	NOBUST <sup>TM</sup>	Starlink <sup>TM</sup>
BIPLAN <sup>TM</sup>	GTO <sup>TM</sup>	NS486 <sup>TM</sup>	STARPLEX <sup>TM</sup>
BLC <sup>TM</sup>	HEX 3000 <sup>TM</sup>	NSC800 <sup>TM</sup>	ST-NIC <sup>TM</sup>
BLX <sup>TM</sup>	HiSeCT <sup>TM</sup>	NSCISE <sup>TM</sup>	SuperAT <sup>TM</sup>
BMAC <sup>TM</sup>	HPC <sup>TM</sup>	NSX-16 <sup>TM</sup>	Super-Block <sup>TM</sup>
Boomer <sup>®</sup>	HyBal <sup>TM</sup>	NSX-C-16 <sup>TM</sup>	SuperChip <sup>TM</sup>
Brite-Lite <sup>TM</sup>	I <sup>3</sup> L <sup>®</sup>	NTERCOM <sup>TM</sup>	SuperI/O <sup>TM</sup>
BSIT <sup>TM</sup>	ICM <sup>TM</sup>	NURAM <sup>TM</sup>	SuperScript <sup>TM</sup>
BSI-2TM	Integral ISET <sup>TM</sup>	OPAL <sup>TM</sup>	<i>Switchers Made Simple<sup>®</sup></i>
CDDT <sup>TM</sup>	Intellisplay <sup>TM</sup>	Overture <sup>TM</sup>	SYS32 <sup>TM</sup>
CDLT <sup>TM</sup>	Inter-LERIC <sup>TM</sup>	OxISS <sup>TM</sup>	TapePak <sup>®</sup>
CGS <sup>TM</sup>	Inter-RIC <sup>TM</sup>	P2CMOST <sup>TM</sup>	TDST <sup>TM</sup>
CIM <sup>TM</sup>	ISE <sup>TM</sup>	Perfect Watch <sup>TM</sup>	TeleGate <sup>TM</sup>
CIMBUST <sup>TM</sup>	ISE/06 <sup>TM</sup>	PLANAT <sup>TM</sup>	The National Anthem <sup>®</sup>
CLASIC <sup>TM</sup>	ISE/08 <sup>TM</sup>	PLANAR <sup>TM</sup>	TinyPaK <sup>TM</sup>
COMBO <sup>®</sup>	ISE/16 <sup>TM</sup>	PLAYER <sup>TM</sup>	TLC <sup>TM</sup>
COMBO I <sup>TM</sup>	ISE32 <sup>TM</sup>	PLAYER + M <sup>TM</sup>	Trapezoidal <sup>TM</sup>
COMBO II <sup>TM</sup>	ISOPLANAR <sup>TM</sup>	PLLatinum <sup>TM</sup>	TRI-CODE <sup>TM</sup>
CompactRISC <sup>TM</sup>	ISOPLANAR-Z <sup>TM</sup>	Plus-2 <sup>TM</sup>	TRI-POLY <sup>TM</sup>
CompactSPEECH <sup>TM</sup>	LERIC <sup>TM</sup>	Polycraft <sup>TM</sup>	TRI-SAFET <sup>TM</sup>
COPST <sup>TM</sup> microcontrollers	LMCMOST <sup>TM</sup>	POPT <sup>TM</sup>	TRI-STATE <sup>®</sup>
COP8 <sup>TM</sup>	M2CMOST <sup>TM</sup>	Power + Control <sup>TM</sup>	TROPIC <sup>TM</sup>
CRDT <sup>TM</sup>	Macrobus <sup>TM</sup>	POWERplanar <sup>TM</sup>	Tropic Pele <sup>TM</sup>
<i>CROSSVOLT<sup>TM</sup></i>	Macrocomponent <sup>TM</sup>	QST <sup>TM</sup>	Tropic Reef <sup>TM</sup>
CSN <sup>1</sup> <sup>TM</sup>	MACSI <sup>TM</sup>	QUAD3000 <sup>TM</sup>	TURBOTRANSCEIVER <sup>TM</sup>
CTI <sup>TM</sup>	MAPL <sup>TM</sup>	Quiet Series <sup>TM</sup>	TWISTER <sup>TM</sup>
CYCLONET <sup>TM</sup>	MAXI-ROM <sup>®</sup>	QUIKLOOK <sup>TM</sup>	VIPT <sup>TM</sup>
DA4 <sup>TM</sup>	Microbus <sup>TM</sup> data bus	RATT <sup>TM</sup>	VR32 <sup>TM</sup>
DENSPAK <sup>TM</sup>	MICRO-DAC <sup>TM</sup>	RIC <sup>TM</sup>	WATCHDOG <sup>TM</sup>
DIB <sup>TM</sup>	μPot <sup>TM</sup>	RICKIT <sup>TM</sup>	XMOS <sup>TM</sup>
DISCERN <sup>TM</sup>	μtalker <sup>TM</sup>	RTX16 <sup>TM</sup>	XPU <sup>TM</sup>
DISTILL <sup>TM</sup>	Microtalker <sup>TM</sup>	SCENIC <sup>TM</sup>	Z STAR <sup>TM</sup>
DNR <sup>®</sup>	MICROWIRE <sup>TM</sup>	SCX <sup>TM</sup>	883B/RET <sup>STM</sup>
DPVM <sup>TM</sup>	MICROWIRE/PLUST <sup>TM</sup>		883S/RET <sup>STM</sup>
E2CMOST <sup>TM</sup>			

ABEL<sup>TM</sup> is a trademark of Data I/O Corp.

Apple<sup>®</sup>, Appletalk<sup>®</sup> and Macintosh<sup>®</sup> are registered trademarks of Apple Corporation.

Dataphone<sup>®</sup> is a registered trademark of Dataphone Digital Service Corporation.

EtherCard PLUS<sup>TM</sup> and EtherCard PLUS 16<sup>TM</sup> are trademarks of Standard Microsystems Corp.

Ethernet<sup>®</sup> is a registered trademark of Xerox Corporation.

GAL<sup>®</sup> is a registered trademark of Lattice Semiconductor Corp.

General Motors<sup>®</sup> is a registered trademark of General Motors Corporation.

IBM<sup>®</sup>, MicroChannel<sup>®</sup>, NetView<sup>®</sup>, OS/2<sup>®</sup>, PC<sup>®</sup>, PC-AT<sup>®</sup>, PC-X<sup>®</sup>, PS/2<sup>®</sup> and SNA<sup>®</sup> are registered trademarks of International Business Machines Corp.

Microsoft<sup>®</sup>, Microsoft Windows for Workgroups<sup>®</sup> and MS<sup>®</sup> are registered trademarks of Microsoft Corporation.

NE2000<sup>plus</sup><sup>TM</sup> and NetWare<sup>TM</sup> are trademarks of Novell, Inc.

Novell<sup>®</sup> is a registered trademark of Novell, Inc.

PAL<sup>®</sup> is a registered trademark of and used under license from Advanced Micro Devices, Inc.

UNIX<sup>®</sup> is a registered trademark of AT&T Bell Laboratories Corporation.

VAX<sup>TM</sup> is a trademark of Digital Equipment Corporation.

Velcro<sup>®</sup> is a registered trademark of Velcro USA, Inc.

3Com<sup>®</sup> is a registered trademark of 3Com Corporation.

## LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

**National Semiconductor Corporation** 2900 Semiconductor Drive, P.O. Box 58090, Santa Clara, California 95052-8090 1-800-272-9959 TWX (910) 339-9240

National does not assume any responsibility for use of any circuitry described, no circuit patent licenses are implied, and National reserves the right, at any time without notice, to change said circuitry or specifications.

## Product Status Definitions

### Definition of Terms

Data Sheet Identification	Product Status	Definition
<b>Advance Information</b>	Formative or In Design	This data sheet contains the design specifications for product development. Specifications may change in any manner without notice.
<b>Preliminary</b>	First Production	This data sheet contains preliminary data, and supplementary data will be published at a later date. National Semiconductor Corporation reserves the right to make changes at any time without notice in order to improve design and supply the best possible product.
<b>No Identification Noted</b>	Full Production	This data sheet contains final specifications. National Semiconductor Corporation reserves the right to make changes at any time without notice in order to improve design and supply the best possible product.
<b>Obsolete</b>	Not In Production	This data sheet contains specifications on a product that has been discontinued by National Semiconductor Corporation. The data sheet is printed for reference information only.

National Semiconductor Corporation reserves the right to make changes without further notice to any products herein to improve reliability, function or design. National does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent rights, nor the rights of others.

# Table of Contents

Alphanumeric Index .....	vi
An Introduction to Local Area Network (LAN) Standards and an Overview of National Semiconductor Products .....	viii
<b>ETHERNET PROTOCOL PRODUCTS</b>	
<b>Section 1 Integrated Network Interface Controller Products</b>	
<b>8/16-Bit 10 Mb/s Network Interface Controllers</b>	
DP83907 AT/LANTIC II .....	1-3
DP83905 AT/LANTIC AT Local Area Network Twisted Pair Interface Controller .....	1-71
DP83902A ST-NIC Serial Network Interface Controller for Twisted Pair .....	1-150
DP83901A SNIC Serial Network Interface Controller .....	1-217
DP8390D/NS32490D NIC Network Interface Controller .....	1-279
AN-475 DP8390 Family Network Interface Controller: An Introductory Guide .....	1-334
AN-686 Ethernet Network Interface Adapter for the Apple Macintosh II NuBus .....	1-342
AN-729 DP839EB-ATN IBM PC-AT Compatible DP83901 SNIC Serial Network Interface Controller Evaluation Board .....	1-372
AN-752 DP83902EB-AT PC-AT Compatible DP83902 ST-NIC Ethernet Evaluation Board .....	1-385
AN-792 NM95C12 Applications in a PC-AT Ethernet Adapter .....	1-400
AN-842 The Design and Operation of a Low Cost 8-Bit PC-XT Compatible Ethernet Adapter Using the DP83902 .....	1-404
AN-858 Guide to Loopback Using the DP8390 Chip Set .....	1-418
AN-873 Architectural Choices for Network Performance .....	1-424
AN-874 Writing Drivers for the DP8390 NIC Family of Ethernet Controllers .....	1-439
AN-875 DP83905EB-AT AT/LANTIC Evaluation Board .....	1-453
AN-886 The Operation of the FIFOs in the DP8390, DP83901, DP83902 and DP83905 .....	1-471
AN-887 AT/LANTIC Software Developer's Guide .....	1-491
AN-892 I/O Channel Ready Considerations for the DP83902EB-AT .....	1-505
AN-897 DP83905EB-AT AT/LANTIC Hardware Users' Guide .....	1-509
AN-973 Layout Guidelines for an Ethernet Adapter Using the DP83907 AT/LANTIC II ..	1-525
AN-974 Design Guide for an Ethernet Adapter Using the DP83907 AT/LANTIC II .....	1-528
<b>16/32-Bit 10 Mb/s Systems-Oriented Network Interface Controller</b>	
DP83934CVUL-20/25 MHz SONIC-T Systems-Oriented Network Interface Controller with Twisted Pair Interface .....	1-535
DP83932C SONIC Systems-Oriented Network Interface Controller .....	1-637
DP83916 SONIC-16 Systems-Oriented Network Interface Controller .....	1-734
AN-691 32-Bit Bus Master Ethernet Interface for the 68030 (Using the Macintosh SE/30) .....	1-828
AN-732 DP839EB-MCS SONIC MicroChannel Ethernet Adapter .....	1-837
AN-745 DP83932 SONIC Bus Operations Guide .....	1-862
AN-746 Software Driver Programmer's Guide for the DP83932 SONIC .....	1-874
AN-747 Determining Arbitration and Threshold Levels in a SONIC Based MicroChannel Adapter .....	1-904
AN-792 NM95C12 Applications in a PC-AT Ethernet Adapter .....	1-911
AN-855 DP83916EB-AT: High Performance AT Compatible Bus Master Ethernet Adapter Card .....	1-915
AN-859 DP83932EB-EISA SONIC/EISA Packet Driver for PC/TCP .....	1-949
AN-877 DP83932EB-EISA SONIC EISA Bus Master Ethernet Adapter .....	1-988
<b>Section 2 10 Mb/s Physical Layer Transceivers and ENDECs</b>	
DP8392C/DP8392C-1 CTI Coaxial Transceiver Interface .....	2-3
Reliability Data Summary for DP8392 .....	2-13

# Table of Contents (Continued)

## **Section 2 10 Mb/s Physical Layer Transceivers and ENDECs** (Continued)

DP83910A CMOS SNI Serial Network Interface .....	2-15
DP8391A/NS32491A SNI Serial Network Interface .....	2-25
AN-442 Ethernet/Cheapernet Physical Layer Made Easy with DP8391/92 .....	2-35
AN-620 Interfacing the DP8392 to 93 $\Omega$ and 75 $\Omega$ Cable .....	2-44
AN-622 Low Power Ethernet with the CMOS DP83910 Serial Network Interface .....	2-47
AN-757 Measuring Ethernet Tap Capacitance .....	2-54
Ethernet Magnetics Vendors for 10BASE-T, 10BASE2 and 10BASE5.....	2-57

## **Section 3 10 Mb/s Repeater Interface Controller Products**

DP83950B RIC Repeater Interface Controller .....	3-3
DP83952 Repeater Interface Controller with Security Features (RIC II) .....	3-82
DP83955A/DP83956A LERIC Lite Repeater Interface Controller .....	3-171
DP83957 10 Mb/s Repeater Information Base .....	3-222
AN-781 DP83950EB-AT IEEE 802.3 Multi-Port Repeater Evaluation Kit .....	3-245
AN-782 RIC-SONIC Interface .....	3-258
AN-783 DP83950 Twisted Pair Parametric Evaluation .....	3-262
AN-843 Introduction to Repeaters, the RIC and LERIC, and Their Applications .....	3-270
AN-854 DP83956EB-AT LERIC (Lite Repeater Interface Controller) PC-AT Adapter ...	3-289
AN-888 RFI Suppression Techniques in DP83950 RIC Based Systems .....	3-308
AN-896 DP83956EB-SA Stand Alone Hub .....	3-319

## **Section 4 100 Mb/s Fast Ethernet Protocol Products**

DP83223A TWISTER High Speed Networking Transceiver Device .....	4-3
DP83840 10/100 Mb/s Ethernet Physical Layer .....	4-21
DP83850 100 Mb/s Repeater Interface Controller .....	4-76
DP83856 100 Mb/s Repeater Information Base .....	4-109
AN-986 An Introduction to Auto-Negotiation .....	4-137
AN-1010 100BASE-TX Unmanaged Repeater Design Recommendations .....	4-144
AN-336 Understanding Integrated Circuit Package Power Capabilities .....	4-169
AN-780 Operating ECL from a Single Positive Supply .....	4-174
Transmission Line Concepts .....	4-181
System Considerations .....	4-194

## **SUPPORT MATERIALS**

### **Section 5 Glossary and Acronyms**

Glossary of Local Area Networking and Data Communications Terms .....	5-3
Ethernet and Networking Acronyms .....	5-17

### **Section 6 Appendix/Physical Dimensions**

Physical Dimensions .....	6-3
Bookshelf	
Distributors	
Worldwide Sales Offices	

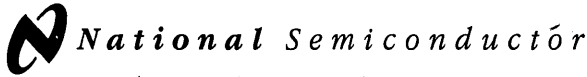
# Alpha-Numeric Index

AN-336 Understanding Integrated Circuit Package Power Capabilities .....	4-169
AN-442 Ethernet/CheaperNet Physical Layer Made Easy with DP8391/92 .....	2-35
AN-475 DP8390 Family Network Interface Controller: An Introductory Guide .....	1-334
AN-620 Interfacing the DP8392 to 93Ω and 75Ω Cable .....	2-44
AN-622 Low Power Ethernet with the CMOS DP83910 Serial Network Interface .....	2-47
AN-686 Ethernet Network Interface Adapter for the Apple Macintosh II NuBus .....	1-342
AN-691 32-Bit Bus Master Ethernet Interface for the 68030 (Using the Macintosh SE/30) .....	1-828
AN-729 DP839EB-ATN IBM PC-AT Compatible DP83901 SNIC Serial Network Interface Controller Evaluation Board .....	1-372
AN-732 DP839EB-MCS SONIC MicroChannel Ethernet Adapter .....	1-837
AN-745 DP83932 SONIC Bus Operations Guide .....	1-862
AN-746 Software Driver Programmer's Guide for the DP83932 SONIC .....	1-874
AN-747 Determining Arbitration and Threshold Levels in a SONIC Based MicroChannel Adapter .....	1-904
AN-752 DP83902EB-AT PC-AT Compatible DP83902 ST-NIC Ethernet Evaluation Board .....	1-385
AN-757 Measuring Ethernet Tap Capacitance .....	2-54
AN-780 Operating ECL from a Single Positive Supply .....	4-174
AN-781 DP83950EB-AT IEEE 802.3 Multi-Port Repeater Evaluation Kit .....	3-245
AN-782 RIC-SONIC Interface .....	3-258
AN-783 DP83950 Twisted Pair Parametric Evaluation .....	3-262
AN-792 NM95C12 Applications in a PC-AT Ethernet Adapter .....	1-911
AN-842 The Design and Operation of a Low Cost 8-Bit PC-XT Compatible Ethernet Adapter Using the DP83902 .....	1-404
AN-843 Introduction to Repeaters, the RIC and LERIC, and Their Applications .....	3-270
AN-854 DP83956EB-AT LERIC (Lite Repeater Interface Controller) PC-AT Adapter .....	3-289
AN-855 DP83916EB-AT: High Performance AT Compatible Bus Master Ethernet Adapter Card .....	1-915
AN-858 Guide to Loopback Using the DP8390 Chip Set .....	1-418
AN-859 DP83932EB-EISA SONIC/EISA Packet Driver for PC/TCP .....	1-949
AN-873 Architectural Choices for Network Performance .....	1-424
AN-874 Writing Drivers for the DP8390 NIC Family of Ethernet Controllers .....	1-439
AN-875 DP83905EB-AT AT/LANTIC Evaluation Board .....	1-453
AN-877 DP83932EB-EISA SONIC EISA Bus Master Ethernet Adapter .....	1-988
AN-886 The Operation of the FIFOs in the DP8390, DP83901, DP83902 and DP83905 .....	1-471
AN-887 AT/LANTIC Software Developer's Guide .....	1-491
AN-888 RFI Suppression Techniques in DP83950 RIC Based Systems .....	3-308
AN-892 I/O Channel Ready Considerations for the DP83902EB-AT .....	1-505
AN-896 DP83956EB-SA Stand Alone Hub .....	3-319
AN-897 DP83905EB-AT AT/LANTIC Hardware Users' Guide .....	1-509
AN-973 Layout Guidelines for an Ethernet Adapter Using the DP83907 AT/LANTIC II .....	1-525
AN-974 Design Guide for an Ethernet Adapter Using the DP83907 AT/LANTIC II .....	1-528
AN-986 An Introduction to Auto-Negotiation .....	4-137
AN-1010 100BASE-TX Unmanaged Repeater Design Recommendations .....	4-144
DP8390D NIC Network Interface Controller .....	1-279
DP8391A SNI Serial Network Interface .....	2-25
DP8392C/DP8392C-1 CTI Coaxial Transceiver Interface .....	2-3
DP83223A TWISTER High Speed Networking Transceiver Device .....	4-3
DP83840 10/100 Mb/s Ethernet Physical Layer .....	4-21
DP83850 100 Mb/s Repeater Interface Controller .....	4-76
DP83856 100 Mb/s Repeater Information Base .....	4-109
DP83901A SNIC Serial Network Interface Controller .....	1-217
DP83902A ST-NIC Serial Network Interface Controller for Twisted Pair .....	1-150

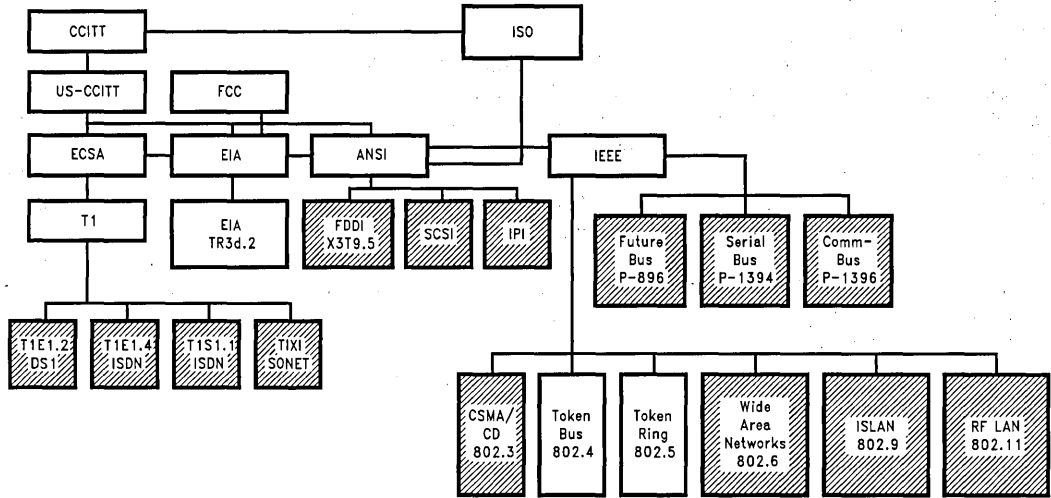


# Alpha-Numeric Index (Continued)

DP83905 AT/LANTIC AT Local Area Network Twisted Pair Interface Controller .....	1-71
DP83907 AT/LANTIC II .....	1-3
DP83910A CMOS SNI Serial Network Interface .....	2-15
DP83916 SONIC-16 Systems-Oriented Network Interface Controller .....	1-734
DP83932C SONIC Systems-Oriented Network Interface Controller .....	1-637
DP83934CVUL-20/25 MHz SONIC-T Systems-Oriented Network Interface Controller with Twisted Pair Interface .....	1-535
DP83950B RIC Repeater Interface Controller .....	3-3
DP83952 Repeater Interface Controller with Security Features (RIC II) .....	3-82
DP83955A LERIC Lite Repeater Interface Controller .....	3-171
DP83956A LERIC Lite Repeater Interface Controller .....	3-171
DP83957 10 Mb/s Repeater Information Base .....	3-222
Ethernet and Networking Acronyms .....	5-17
Ethernet Magnetics Vendors for 10BASE-T, 10BASE2 and 10BASE5 .....	2-57
Glossary of Local Area Networking and Data Communications Terms .....	5-3
NS32490D NIC Network Interface Controller .....	1-279
NS32491A SNI Serial Network Interface .....	2-25
Reliability Data Summary for DP8392 .....	2-13
System Considerations .....	4-194
Transmission Line Concepts .....	4-181



## An Introduction to Local Area Network (LAN) Standards and an Overview of National Semiconductor Products



Note: See Glossary for Definitions of the various main organizations.

TL/F/11257-1

**FIGURE 1. National Semiconductor's Participation In Industry Standards Committees**

### PRO-ACTIVE COMMUNICATIONS STANDARDS INVOLVEMENT

Standards are important in the development and growth of the communications industry. Several international committees and organizations define and promote standards to ensure interoperability and interconnectivity among multiple vendors' products.

National Semiconductor takes an active part in the development and definition of these standards (see Figure 1) and

produces a wide range of products that meet and exceed the requirements of these standards.

National's data communications portfolio includes complete silicon solutions for IEEE 802.3 4th Edition Ethernet, IEEE 802.3u Fast Ethernet, and ANSI X3T9.5 FDDI applications—the broadest portfolio in the industry. See Table I for a protocol characteristics summary.

**TABLE I. Key Characteristics of Popular LAN Protocols**

Characteristic	Ethernet	Fast Ethernet	FDDI
<b>Bandwidth</b>	10 Mbps	100 Mbps	100 Mbps
<b>Topology</b>	Bus, Star	Star	Dual Ring, Star
<b>Media</b>	Twisted Pair, Coax Optical Fiber	Twisted Pair Optical Fiber	Optical Fiber Twisted Pair
<b>Network Access</b>	CSMA/CD	CSMA/CD	Timed-Token-Passing
<b>Frame Size (bytes)</b>	1,500	1,500	4,500
<b>Encoding</b>	Manchester	4B/5B, MLT3	4B /5B
<b>Max. No. Nodes</b>	1,024	1,024	1000
<b>Distance (nodes)</b>	2.8 km	200m	2-3 km
<b>Max. Network Span</b>	2.8 km	2 km	200 km

**TABLE II. IEEE 802.3 Physical Layer Specifications, 10 Mbps Ethernet**

Parameter	10BASE5	10BASE2	10BASE-T
<b>Designator</b>	Thick Coax	Thin Coax	Twisted Pair
<b>Segment Length</b>	500 Meters	185 Meters	100 Meters Nominal
<b>Topology</b>	Bus (Multi-Point)	Bus (Multi-Point)	Star (Point-to-Point)
<b>Cable Type</b>	0.4" Diam. 50 $\Omega$ , Double Shield Coax (RG11)	0.2" Diam. 50 $\Omega$ , Single Shield Coax (RG58)	24 Gauge, 100 $\Omega$ Twisted Pair
<b>Connection</b>	Precision TAP	BNC "T"	8-Pin, RJ-45

**ETHERNET: IEEE 802.3 4th EDITION**

Ethernet is the most widely installed LAN standard for connecting personal computers and workstations with information resources, servers and other peripherals.

The 10 Mbps Ethernet CSMA/CD (Carrier Sense Multiple Access with Collision Detection) protocol defines how a node will gain access to the network. The node first monitors the media to ensure that no transmissions are in progress (Carrier Sense). The node may then decide to transmit (Multiple Access). If more than one node decides to transmit simultaneously, then a collision will occur.

All nodes must be able to detect this condition (Collision Detection), stop their transmissions, and retry after a random period of time.

**Physical Layer Specifications**

Physical Layer Specifications define various media and topology options. Current specifications include 10BASE5, which defines the use of thick, double-shielded coax in a bus configuration; 10BASE2, which defines the use of thin coax in a bus configuration; 10BASE-T, which defines the use of unshielded twisted pair cable in a star configuration; and 10BASE-F, which defines the use of optical fiber in a star configuration. Table II summarizes the specifications for the coax and twisted pair cables.

Because of its topology, 10BASE-T Ethernet LANs require both nodes and hubs or repeaters to extend the signals around the network. Network management becomes an issue or requirement in 10BASE-T as well. The IEEE 802.3 Repeater Specification defines basic, mandatory and optional management attributes to ensure interoperability in repeaters.

**Comprehensive Ethernet Solutions**

National's Ethernet products are grouped into four families: Network Interface Controllers (NIC Family); Systems Oriented Network Interface Controllers (SONIC™ Family); Physical Layers and Encoders/Decoders (PHYs and ENDECs Family); and Repeater Interface Controllers (RIC™ Family).

The NIC Family of 8/16-bit controllers provides value-based solutions for the node environment and designs. It consists of the DP8390 NIC, DP83901 SNIC™, DP83902 ST-NIC™, and DP83905 AT/LANTIC™. All are based on the industry standard DP8390 NIC core. The AT/LANTIC, the most highly integrated of the family, has an AT Bus Interface and twisted pair transceiver integrated on-chip making it well suited for motherboard designs. The ST-NIC also has an integrated twisted pair transceiver, but has no on-chip bus interface making it ideal for specialty I/O applications, such as PCMCIA, and non-AT buses. The SNIC provides a MAC and ENDEC only. The NIC is only a MAC.

The SONIC Family of 16/32-bit controllers provides full performance solutions for nodes and interconnect products. It consists of the DP83932 SONIC, DP83934 SONIC-T, and DP83916 SONIC-I6. The SONIC is appropriate for 16- and

32-bit Ethernet network nodes and bridges for any of the standard media choices. The SONIC-T is ideal for 10BASE-T networks due to its integrated twisted pair transceiver. The SONIC-16 is for 16-bit designs using the standard media choices.

The PHYs and ENDECs consist of separate physical layer transceivers and encoder/decoders. The DP8392 CTI™ is the industry's most reliable and most popular transceiver. It is often used in Media Access Units.

The RIC Family of repeater devices offers solutions for both high and low end hubs. The DP83950 RIC provides full management capabilities for large, multi-port, expandable hubs while the DP83955/56 LERIC™ offers limited management for small, basic functionality hubs.

More detailed information, including specific datasheets and application notes for these products, are included in Sections 1-4 of this book.

**FAST ETHERNET: IEEE 802.3u**

Fast Ethernet is another LAN standard that is similar to ordinary Ethernet, but ten times faster: 100 Mbps rather than 10 Mbps. Like ordinary Ethernet, it uses CSMA/CD and offers a choice of cable types.

**Physical Layer Specifications**

The current Physical Layer Specifications for Fast Ethernet include 100BASE-TX, which defines the use of two unshielded twisted pair cables; 100BASE-FX, which defines the use of optical fiber; and 100BASE-T4, which defines the use of four unshielded twisted pair cables. These cabling options operate in a star (point-to-point) configuration. Table III summarizes these Physical Layer Specifications.

**Comprehensive Fast Ethernet Solutions**

National's Fast Ethernet products provide a wide range of support for implementation of Fast Ethernet LAN's.

The DP83223 TWISTER™ High Speed Networking Transceiver Device is a twisted pair transceiver designed to operate in either a Fast Ethernet 100BASE-TX or FDDI LAN environment.

The DP83840 Ethernet Physical Layer is a device designed for easy implementation of Ethernet or Fast Ethernet using unshielded, shielded, or fiber optic cables.

The DP83810 Ethernet Controller for the Peripheral Component Interconnect Local Bus provides an Ethernet or Fast Ethernet interface for computer systems using the PCI Local Bus. For Fast Ethernet applications, it provides a complete solution when used with the DP83840 Ethernet Physical Layer and the DP83223 High Speed Networking Transceiver.

The DP83856 Repeater Information Base, used together with one or more DP83850 Repeater Interface Controller devices, simplifies the design of managed multipoint Fast Ethernet repeaters. Each DP83850 directly supports up to 12 network connections.

TABLE III. IEEE 802.3u Physical Layer Specifications, 100 Mbps Ethernet

Parameter	100BASE-TX	100BASE-FX	100BASE-T4
<b>Designator</b>	Twisted Pair	Fiber	Twisted Pair
<b>Segment Length</b>	100 Meters	114 Meters*	100 Meters
<b>Topology</b>	Star (Point-to-Point)	Star (Point-to-Point)	Star (Point-to-Point)
<b>Required Cabling</b>	2 Twisted Pairs	1 Multi-Mode Fiber or 2 Single-Made Fibers	4 Twisted Pairs
<b>Cable Quality</b>	Category 5 Unshielded Twisted Pair (UTP) or Shielded Twisted Pair (STP)	N/A	Category 3, 4 or 5 Unshielded Twisted Pair (UTP)
<b>Cable Type</b>	24 Gauge 100Ω UTP or 150Ω STP	62.5/125 μm Multi-Mode or Single-Mode Fiber	24 Gauge, 100Ω UTP
<b>Connection</b>	8-pin, RJ-45 (for UTP) or 9-pin, D8-9 (for STP)	Low Cost Fiber Optical Interface Connector (duplex SC), Media Interface Connector (MIC), or Optical Medium Connector Plug and Socket (ST)	8-pin, RJ-45

\*100BASE-FX segment length is limited by the collision domain of Ethernet. Longer segment lengths (i.e., 2 km or greater) are possible when used with switched Ethernet devices.

**FDDI (Fiber Distributed Data Interface): ANSI X3T9.5**

Fiber Distributed Data Interface (FDDI) is a 100 Mbps networking standard that uses a token passing access method and dual rings of optical fiber or twisted pair cable.

Like Token Ring, FDDI utilizes a ring topology. However, FDDI has a dual-ring configuration providing greater fault tolerance and redundancy. Timed-token-passing for FDDI works in a manner similar to token-passing for Token Ring. The standard defines fiber as the physical media, but a low cost physical media specification is in development to standardize the use of shielded and unshielded twisted pair media.

Originally used for backbone applications to connect multiple islands of lower speed LANs, FDDI has since migrated to the desktop. The inherent management capabilities and use of fiber offers a means with which to expand networks in a controlled manner. The use of twisted pair media for FDDI products provides a cost-effective means of bringing high-speed networking to the desktop.

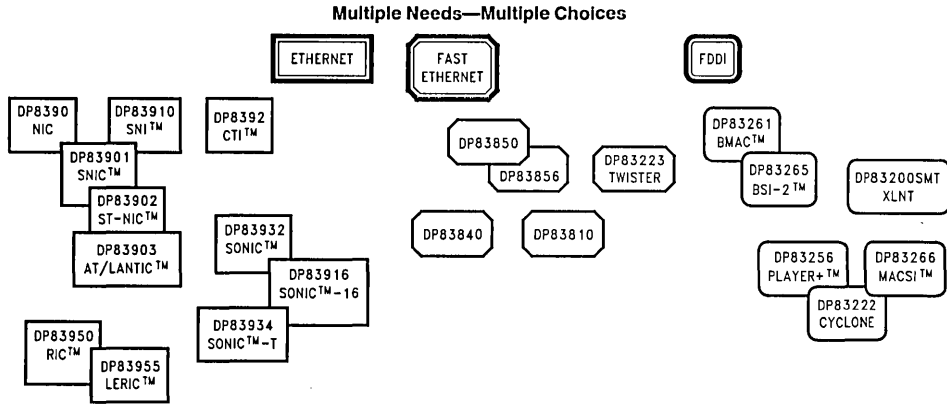
National's FDDI DP83200 Series portfolio consists of a two chip set, a five chip set, and a MLT-3/NRZI transceiver for UTP/STP cabling. A brief overview of these products is included in Section 6 for your reference. More detailed infor-

mation is available in the 1995 FDDI Databook or through your local National Semiconductor Sales Office.

**National's Commitment to Networking Standards**

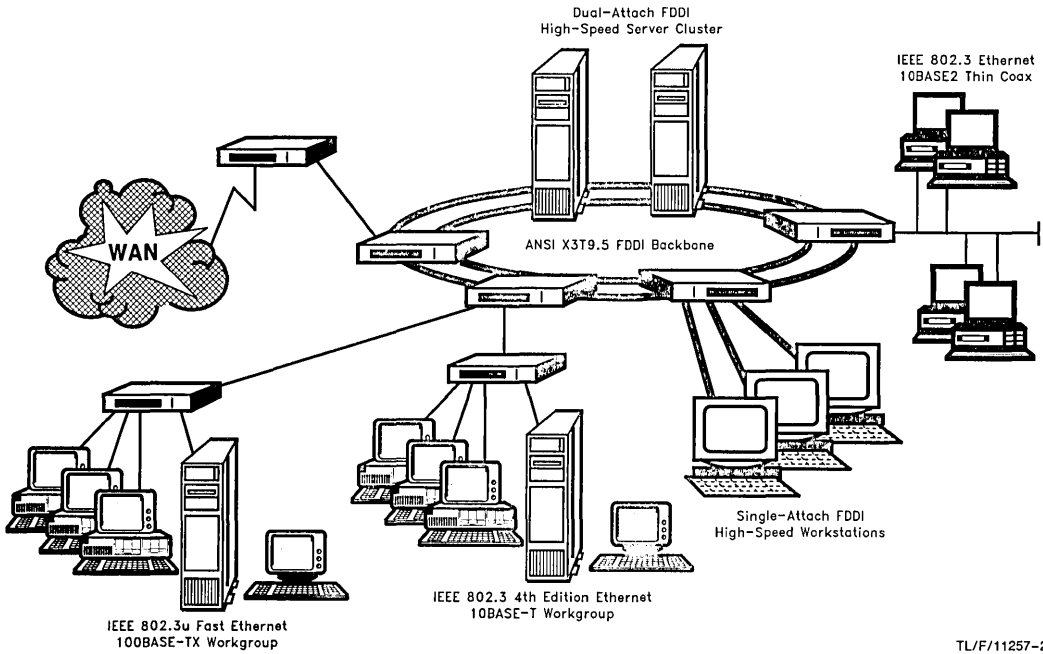
In support of current networking standards, National Semiconductor provides an extensive selection of networking silicon for Ethernet, Fast Ethernet, and FDDI. Comprehensive evaluation platforms, demonstration software, application notes, and system briefs are also available to provide network product designers and manufactures optimal flexibility and ease of product development for today's networking needs.

National Semiconductor's active involvement in the development of communications standards and experience as a LAN user have resulted in the availability of a broad range of silicon solutions for the networking market. National understands the importance of preserving investments in existing standards while balancing participation in the innovation and development of emerging networking standards such as isochronous Ethernet, ATM, and FDDI-II. With continuing efforts on standards committees, product innovation, and market leadership, National is dedicated to serving the networking needs today and in the future.



**FIGURE 2. National Semiconductor LAN Silicon Solutions**

TL/F/11257-3



**FIGURE 3. Mixed Protocol LAN Environment**

TL/F/11257-2





## Section 1

# ETHERNET PROTOCOL PRODUCTS

## **Integrated Network Interface Controller Products**



## Section 1 Contents

### 8/16-Bit 10 Mb/s Network Interface Controllers

DP83907 AT/LANTIC II .....	1-3
DP83905 AT/LANTIC AT Local Area Network Twisted Pair Interface Controller .....	1-71
DP83902A ST-NIC Serial Network Interface Controller for Twisted Pair .....	1-150
DP83901A SNIC Serial Network Interface Controller .....	1-217
DP8390D/NS32490D NIC Network Interface Controller .....	1-279
AN-475 DP8390 Family Network Interface Controller: An Introductory Guide .....	1-334
AN-686 Ethernet Network Interface Adapter for the Apple Macintosh II NuBus .....	1-342
AN-729 DP839EB-ATN IBM PC-AT Compatible DP83901 SNIC Serial Network Interface Controller Evaluation Board .....	1-372
AN-752 DP83902EB-AT PC-AT Compatible DP83902 ST-NIC Ethernet Evaluation Board ....	1-385
AN-792 NM95C12 Applications in a PC-AT Ethernet Adapter .....	1-400
AN-842 The Design and Operation of a Low Cost 8-Bit PC-XT Compatible Ethernet Adapter Using the DP83902 .....	1-404
AN-858 Guide to Loopback Using the DP8390 Chip Set .....	1-418
AN-873 Architectural Choices for Network Performance .....	1-424
AN-874 Writing Drivers for the DP8390 NIC Family of Ethernet Controllers .....	1-439
AN-875 DP83905EB-AT AT/LANTIC Evaluation Board .....	1-453
AN-886 The Operation of the FIFOs in the DP8390, DP83901, DP83902 and DP83905 .....	1-471
AN-887 AT/LANTIC Software Developer's Guide .....	1-491
AN-892 I/O Channel Ready Considerations for the DP83902EB-AT .....	1-505
AN-897 DP83905EB-AT AT/LANTIC Hardware Users' Guide .....	1-509
AN-973 Layout Guidelines for an Ethernet Adapter Using the DP83907 AT/LANTIC II .....	1-525
AN-974 Design Guide for an Ethernet Adapter Using the DP83907 AT/LANTIC II .....	1-528

### 16/32-Bit 10 Mb/s Systems-Oriented Network Interface Controller

DP83934CVUL-20/25 MHz SONIC-T Systems-Oriented Network Interface Controller with Twisted Pair Interface .....	1-535
DP83932C SONIC Systems-Oriented Network Interface Controller .....	1-637
DP83916 SONIC-16 Systems-Oriented Network Interface Controller .....	1-734
AN-691 32-Bit Bus Master Ethernet Interface for the 68030 (Using the Macintosh SE/30) ....	1-828
AN-732 DP839EB-MCS SONIC MicroChannel Ethernet Adapter .....	1-837
AN-745 DP83932 SONIC Bus Operations Guide .....	1-862
AN-746 Software Driver Programmer's Guide for the DP83932 SONIC .....	1-874
AN-747 Determining Arbitration and Threshold Levels in a SONIC Based MicroChannel Adapter .....	1-904
AN-792 NM95C12 Applications in a PC-AT Ethernet Adapter .....	1-911
AN-855 DP83916EB-AT: High Performance AT Compatible Bus Master Ethernet Adapter Card .....	1-915
AN-859 DP83932EB-EISA SONIC/EISA Packet Driver for PC/TCP .....	1-949
AN-877 DP83932EB-EISA SONIC EISA Bus Master Ethernet Adapter .....	1-988



## DP83907 AT/LANTIC™ II

### General Description

The DP83907 Twisted-Pair Enhanced Coaxial Network Interface Controller is a CMOS VLSI device designed for easy implementation of CSMA/CD local area networks.

Unique to the DP83907 is the integration of the entire bus interface for PCAT ISA (Industry Standard Architecture) bus based systems. Hardware and software selectable options allow the DP83907's bus interface to be configured in the same manner as an NE2000 Architecture. All bus drivers and control logic are integrated to reduce board cost and area.

Supported network interfaces include 10BASE5 or 10BASE2 Ethernet via an external transceiver connected to its AUI port, and Twisted-pair Ethernet® (10BASE-T) using the on-board transceiver. The DP83907 provides the Ethernet Media Access Control (MAC), Encode-Decode (ENDEC) with an AUI interface, and 10BASE-T transceiver functions in accordance with the IEEE 802.3 standards.

The DP83907's integrated 10BASE-T transceiver fully complies with the IEEE standard. This functional block incorporates the receiver, transmitter, collision, heartbeat, loop-back, jabber, and link integrity blocks as defined in the standard. The transceiver when combined with equalization resistors, transmit/receive filters, and pulse transformers provides a complete physical interface from the DP83907's ENDEC module and the twisted pair medium. (continued)

### Features

- Controller and integrated bus interface solution for IEEE 802.3, 10BASE5, 10BASE2, and 10BASE-T
- Software compatible with industry standard Ethernet Adapters: Novell®'s NE2000
- No external bus logic or drivers needed
- Supports jumpered or jumperless configuration
- Provides EEPROM interface for non-volatile storage of configuration data, user-defined data and Ethernet Physical Address
- Allows in-situ programming of EEPROM
- Integrated controller, ENDEC, and transceiver
- Full IEEE 802.3 compliant AUI interface
- Single 5V supply

#### 10BASE-T TRANSCIEVER MODULE:

- Integrates transceiver functionality
- Transmitter and receiver functions
- Collision detect, heartbeat and jabber
- Selectable link integrity test or link disable
- Polarity Detection/Correction
- Auto switch
- On chip filter

#### ENDEC MODULE:

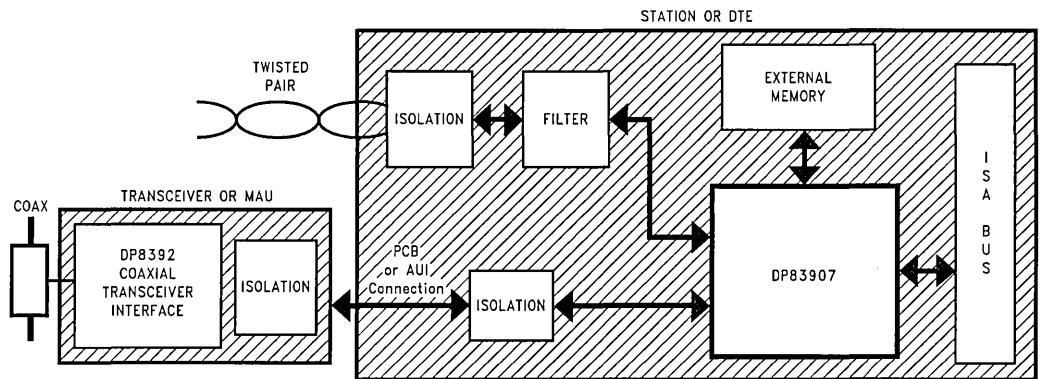
- 10 Mbit/s Manchester encoding/decoding
- Squelch on receive and collision pairs

#### MAC/CONTROLLER MODULE:

- Software compatible with DP8390, DP83901, DP83902
- Efficient buffer management implementation

#### IN-CIRCUIT TEST

## 1.0 System Diagram



TL/F/12082-1

## General Description (Continued)

The integrated ENDEC module allows Manchester encoding and decoding via a differential transceiver and phase lock loop decoder at 10 Mbit/sec. Also included are a collision detect translator and diagnostic loopback capability. The ENDEC module interfaces directly to the transceiver module, and also provides a fully IEEE compliant AUI (Attachment Unit Interface) for connection to other media transceivers.

The Media Access Control function which is provided by the Network Interface Control module (NIC) provides simple and efficient packet transmission and reception control by means of off-board memory which can be accessed through an I/O port.

The DP83907 provides a comprehensive solution for 10BASE-T IEEE 802.3 networks. Due to the inherent constraints of CMOS processing, isolation is required at the AUI differential signal interface for 10BASE5 and 10BASE2 applications.

## Table Of Contents

### 1.0 SYSTEM DIAGRAM

- 1.1 Connection Diagram

### 2.0 PIN DESCRIPTION

### 3.0 SIMPLIFIED APPLICATION DIAGRAM

### 4.0 FUNCTIONAL DESCRIPTION

- 4.1 Bus Interface Block
- 4.2 Power on RESET Operation
- 4.3 EEPROM Operation
- 4.4 Jumperless Operation Support
- 4.5 Ethernet Cable Configuration
- 4.6 Interrupt AND LED Operation
- 4.7 Boot PROM Operation
- 4.8 DP8390 Core (Network interface Controller)
- 4.9 Twisted Pair Interface Module
- 4.10 Encoder/Decoder (ENDEC) Module

### 5.0 REGISTER DESCRIPTIONS

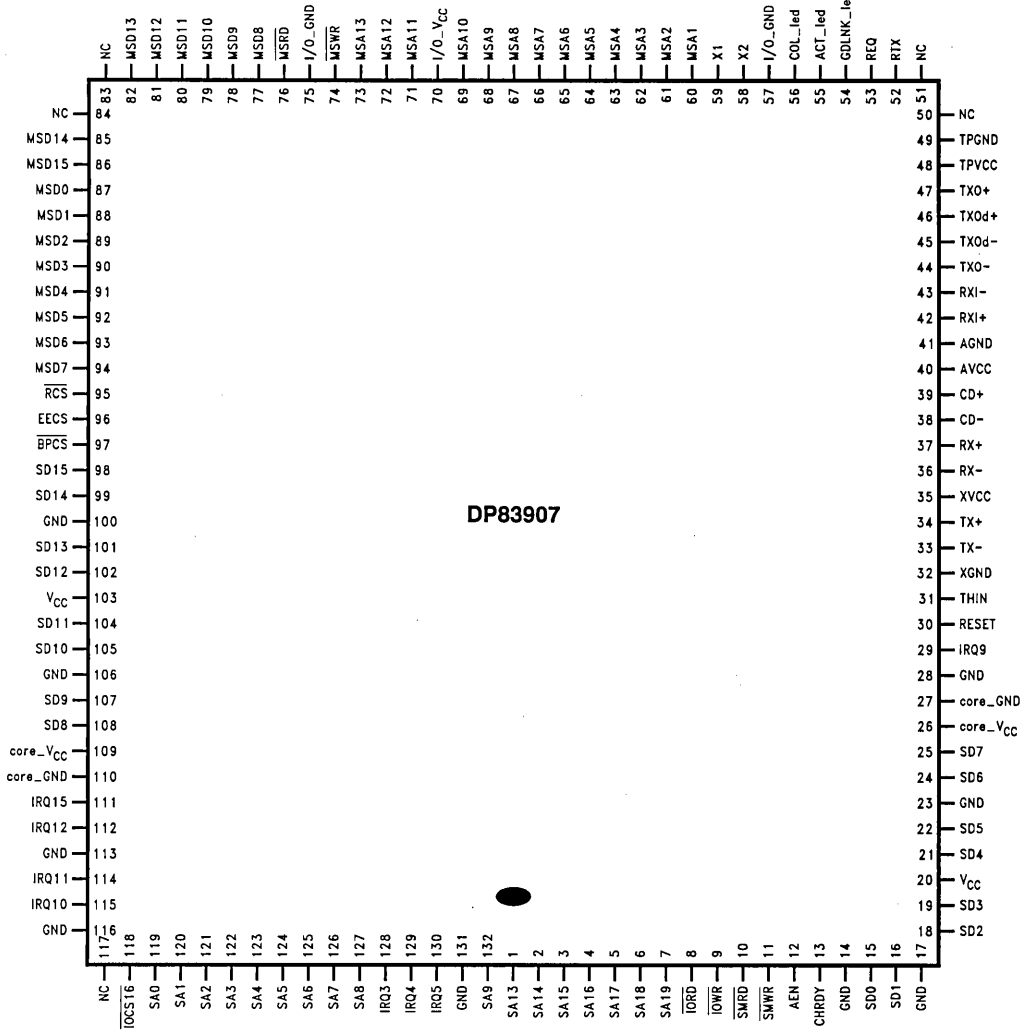
- 5.1 Configuration Registers
- 5.2 NIC Core Registers
- 5.3 DP8390 Core DMA Registers

### 6.0 OPERATION OF DP83907

- 6.1 Transmit/Receive Packet Encapsulation/Decapsulation
- 6.2 Buffer Memory Access Control (DMA)
- 6.3 Packet Reception
- 6.4 Packet Transmission
- 6.5 Loopback Diagnostics
- 6.6 Memory Arbitration and Bus Operation
- 6.7 Functional Bus Timing

### 7.0 IN-CIRCUIT TEST (ICT) DESCRIPTION

# Connection Diagram



Order Number DP83907VF  
See NS Package Number VF132A

TL/F/12082-2



## 2.0 Pin Description

Pin No.	Pin Name	Type	Description
<b>ISA BUS INTERFACE PINS</b>			
119–127, 132	SA0–SA9	I TTL	<b>LATCHED ADDRESS BUS:</b> Low-order 10 bits of the system's latched 20-bit address bus. These bits are used to decode accesses to the DP83907's I/O map.
1–7	SA13–SA19	I TTL	<b>LATCHED ADDRESS BUS:</b> 7 bits of the system's latched 20-bit address bus that are used to decode accesses to the DP83907's boot PROM.
10, 11	$\overline{\text{SMRD}}$ , $\overline{\text{SMWR}}$	I TTL	<b>BOOT PROM STROBES:</b> These inputs are used to access the BOOT PROM.
15, 16, 18 19, 21, 22 24, 25, 98 99, 101, 102, 104, 105, 107, 108	SD0–SD7 SD8–SD15	I/O 3ST	<b>SYSTEM DATA BUS:</b> 16-bit system data bus. Used to transfer data between the system and the DP83907.
118	$\overline{\text{IOCS16}}$	O OCH	<b>16 BIT I/O TRANSFER:</b> This signal indicates that the DP83907 is responding to a 16-bit I/O access by driving 16 bits of data on the bus.
9	$\overline{\text{IOWR}}$	I TTL	<b>I/O WRITE STROBE:</b> Strobe from system to write to the DP83907's I/O map.
8	$\overline{\text{IORD}}$	I TTL	<b>I/O READ STROBE:</b> Strobe from system to read from the DP83907's I/O map.
30	RESET	I TTL	<b>RESET:</b> This signal is output by the system to reset all devices on the bus.
13	CHRDY	O OCH	<b>CHANNEL READY:</b> This signal is used to insert wait states into system accesses.
12	AEN	I TTL	<b>DMA ACTIVE:</b> This signal indicates that the systems DMA has control of the bus.
29, 111, 112, 114, 115, 128 129, 130	IRQ3, 4, 5, 9, 10, 11, 12, 15	O 3ST	<b>INTERRUPT REQUEST:</b> The operation of these outputs is determined by Configuration Register A.

**Note:** Driver Types are: I = Input, O = Output, I/O = Bi-directional Output, OCH = Open Collector, 3ST = TRI-STATE® Output, TTL = TTL Compatible, AUI = Attachment Unit Interface, TPI = Twisted Pair Interface, LED = LED Drive, MOS = CMOS Level Compatible, XTAL = Crystal.

## 2.0 Pin Description (Continued)

Pin No.	Pin Name	Type	Description
<b>NETWORK INTERFACE PINS</b>			
44-47	TxOd+, TXO-, TXO+ TXOd-	O TPI	<b>TWISTED PAIR TRANSMIT OUTPUTS:</b> These high drive CMOS level outputs are resistively combined external to the chip to produce a differential output signal with equalization to compensate for Intersymbol Interference (ISI) on the twisted pair medium.
42, 43	RXI+, RXI-	I TPI	<b>TWISTED PAIR RECEIVE INPUTS:</b> These inputs feed a differential amplifier which passes valid data to the ENDEC module.
33, 34	TX- TX+	O AUI	<b>AUI TRANSMIT OUTPUT:</b> Differential driver which sends the encoded data to the transceiver.
36, 37	RX- RX+	I AUI	<b>AUI RECEIVE INPUT:</b> Differential receive input pair from the transceiver.
38, 39	CD- CD+	I AUI	<b>AUI COLLISION INPUT:</b> Differential collision pair input from the transceiver.
55	ACT_led	O LED	<b>ACTIVITY:</b> An open-drain active low output. It is asserted for approximately 50 ms whenever the DP83907 transmits or receives data in either AUI or TPI modes.
56	COL_led	O LED	<b>COLLISION:</b> An open-drain active low output. It is asserted for approximately 50 ms whenever the DP83907 detects a collision in either either AUI or TPI modes.
54	GDLNK_led	O LED	<b>GOOD LINK:</b> An open-drain active low output. This pin operates as an output to display link integrity status if this function has not been disabled by the GDLNK bit in Configuration Register B. This output is off if the DP83907 is in AUI mode or if link testing is enabled and the link integrity is bad (i.e., the twisted pair link has been broken). This output is on if the DP83907 is in Twisted Pair interface (TPI) mode, link integrity checking is enabled and the link integrity is good (i.e., the twisted pair link has <b>not</b> been broken) or if the link testing is disabled.
53	REQ	I	<b>EQUALIZATION RESISTOR:</b> A resistor can be connected from this pin to GND or V <sub>CC</sub> to change the equalization of the TP output.
52	RTX	I	<b>TRANSMIT LEVEL RESISTOR:</b> A resistor can be connected from this pin to GND or V <sub>CC</sub> to change the TP output amplitude level.
59	X1 (OSCIN)	I XTAL	<b>CRYSTAL ON EXTERNAL OSCILLATOR INPUT</b>
58	X2 (OSCOU)	O XTAL	<b>CRYSTAL FEEDBACK OUTPUT:</b> Used in crystal connections only. Should be left unconnected when using an oscillator module.
31	THIN	O DCDC	<b>THIN CABLE:</b> This output is high if DP83907 is configured for thin cable. It can be used to enable the DC-DC converter required by the thin Ethernet configuration.

**Note:** Driver Types are: I = Input, O = Output, I/O = Bi-directional Output, OCH = Open Collector, 3ST = TRI-STATE Output, TTL = TTL Compatible, AUI = Attachment Unit Interface, TPI = Twisted Pair Interface, LED = LED Drive, MOS = CMOS Level Compatible, XTAL = Crystal.

## 2.0 Pin Description (Continued)

Pin No.	Pin Name	Type	Description
<b>EXTERNAL MEMORY SUPPORT</b>			
87-94	MSD0-7 or CA0-7 or DO, DI, SK	I/O-I-O MOS	<b>MEMORY SUPPORT DATA BUS—CONFIGURATION REGISTER A INPUT: EEPROM SIGNALS:</b> MSD0-7: When RESET is inactive these pins are used to access external memory and boot PROM. CA0-7: When RESET is active Configuration Register A is loaded with the data value on these pins. If the user puts an external pull-up on any of these pins then the corresponding register bit is set to a 1. If the pin is left unconnected then the register bit is 0. DO, DI, SK: When RESET goes from an active to an inactive level DP83907 will read the contents of a serial EEPROM, using these signals, and load the contents into internal registers. These internal registers are mapped into the space taken up by the PROM in the NE2000 Architecture. After the EEPROM read operation has completed these pins will revert to MSD0-2 (D0 = MSD0, DI = MSD1, SK = MSD2).
77-82 85, 86	MSD8-15 or CB0-7	I/O-I MOS	<b>MEMORY SUPPORT DATA BUS—CONFIGURATION REGISTER B INPUT:</b> MSD8-15: When RESET is inactive these pins are used to access external memory. CB0-7: When RESET is active Configuration Register B is loaded with the data value on these pins. If the user puts an external pull-up on any of these pins then the corresponding register bit is set to a 1. If the pin is left unconnected then the register bit is 0.
60-67	MSA1-8 or CC0-7	O-I MOS	<b>MEMORY SUPPORT ADDRESS BUS—CONFIGURATION REGISTER C INPUT:</b> MSA1-8: When RESET is inactive these pins drive the memory support address bus. CC0-7: When RESET is active Configuration Register C is loaded with the data value on these pins. If the user puts an external pull-up on any of these pins then the corresponding register bit is set to a 1. If the pin is left unconnected then the register bit is 0.
68, 69 71-73	MSA9-13 or DWID, EECONFIG and SIG 5-7	O-I MOS	<b>MEMORY SUPPORT ADDRESS BUS—DWID, EECONFIG AND SIGNATURE REGISTER:</b> MSA9-13: When RESET is inactive these pins drive the memory support address bus. DWID (MSA9): When RESET is active this input specifies whether the DP83907 is interfacing to an 8-bit or 16-bit ISA bus. If the user puts an external pull-up on this pin then the bus is considered to be 16-bit. If the pin is left unconnected then the bus is considered to be 8-bit. EECONFIG(MSA10): When RESET is active this input specifies whether the DP83907 loads the configuration from an EEPROM or from the MSD0-15 and MSA1-8 pins. If the user puts an external pull-up on this pin then configuration data is loaded from the EEPROM. If the pin is left unconnected then configuration data is loaded from the memory support bus. SIG 5-7(MSA11-13): When RESET is active the most significant 3 bits of the signature register are loaded with the data value on these pins. If the user puts an external pull-up on any of these pins then the corresponding register bit is set to a 1. If the pin is left unconnected then the register bit is 0.
76	MSRD	O MOS	<b>MEMORY SUPPORT BUS READ:</b> Strobes data from the external RAM into the DP83907 via the memory support data bus.
74	MSWR	O MOS	<b>MEMORY SUPPORT BUS WRITE:</b> Strobes data from the DP83907 into the external RAM via the memory support data bus.

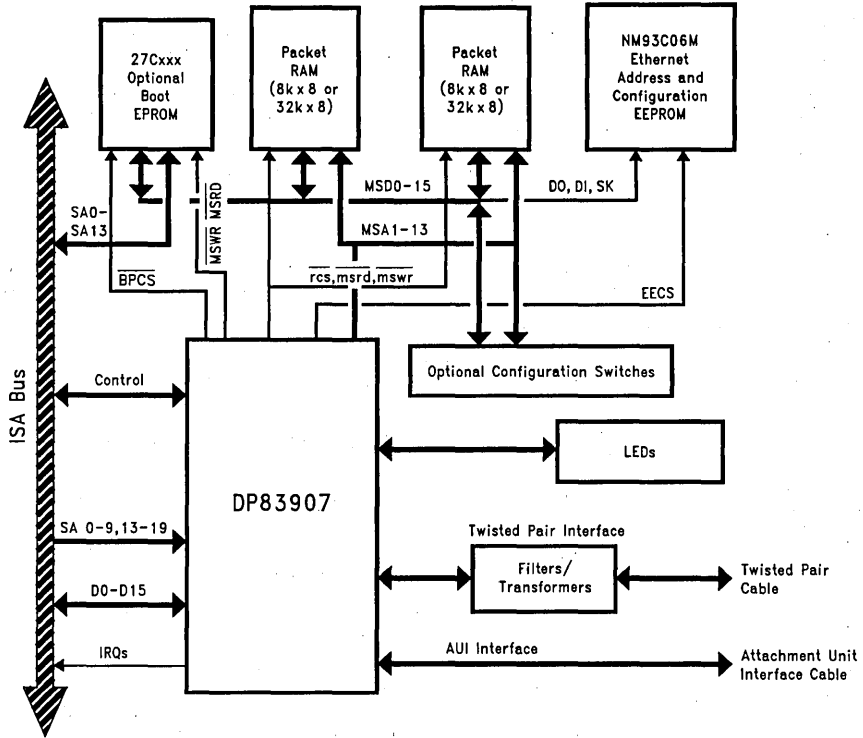
**Note:** Driver Types are: I = Input, O = Output, I/O = Bi-directional Output, OCH = Open Collector, 3ST = TRI-STATE Output, TTL = TTL Compatible, AUI = Attachment Unit Interface, TPI = Twisted Pair Interface, LED = LED Drive, MOS = CMOS Level Compatible, XTAL = Crystal.

## 2.0 Pin Description (Continued)

Pin No.	Pin Name	Type	Description
<b>EXTERNAL MEMORY SUPPORT (Continued)</b>			
95	RCS	O MOS	<b>RAM CHIP SELECT:</b> Drives the chip select of the external RAM.
97	BPCS	O MOS	<b>BOOT PROM CHIP SELECT:</b> Selects the boot PROM on the memory support data bus. This is for READ only.
96	EECS	O MOS	<b>EEPROM CHIP SELECT:</b> Strobes data from the EEPROM onto the memory support data bus.
<b>POWER SUPPLY PINS</b>			
40	AV <sub>CC</sub>		<b>ANALOG 5V SUPPLY PIN:</b> This pin supplies 5V to the DP83907's analog circuitry. To maximize data recovery it is recommended that analog layout and decoupling rules be applied between this pin and AGND.
41	AGND		<b>ANALOG NEGATIVE (GROUND) SUPPLY PIN.</b>
35, 48	XV <sub>CC</sub> , TPV <sub>CC</sub>		<b>PHYSICAL MEDIA 5V SUPPLY PINS:</b> This pin supplies 5V to the DP83907's analog physical media interface circuitry.
32, 49	XGND, TPGND		<b>PHYSICAL MEDIA NEGATIVE (GROUND) SUPPLY PINS:</b> This pin is the ground to the DP83907's analog physical media interface circuitry.
26, 109	coreV <sub>CC</sub>		<b>POSITIVE 5V SUPPLY PINS:</b> These pins supply power to the DP83907.
27, 110	coreGND		<b>NEGATIVE (GROUND) SUPPLY PINS:</b> These are the supply pins for the DP83907. It is suggested that decoupling capacitors be connected between the V <sub>CC</sub> and GND pins. It is essential to provide a path to ground for the GND pins with the lowest possible impedance.
20, 70, 103	V <sub>CC</sub>		<b>POSITIVE 5V SUPPLY PINS:</b> These pins supply power to the DP83907 Output Drivers e.g., SD, MSD, MSA, Chip selects.
14, 17, 23, 28, 57, 75, 100, 106, 113, 116, 131	GND		<b>NEGATIVE (GROUND) SUPPLY PINS:</b> These are the supply pins for the DP83907 Drivers. It is suggested that decoupling capacitors be connected between the V <sub>CC</sub> and GND pins. It is essential to provide a path to ground for the GND pins with the lowest possible impedance.

**Note:** Driver Types are: I = Input, O = Output, I/O = Bi-directional Output, OCH = Open Collector, 3ST = TRI-STATE Output, TTL = TTL Compatible, AUI = Attachment Unit Interface, TPI = Twisted Pair Interface, LED = LED Drive, MOS = CMOS Level Compatible, XTAL = Crystal.

### 3.0 Simplified Application Diagram



TL/F/12082-3



### 3.0 Simplified Application Diagram (Continued)

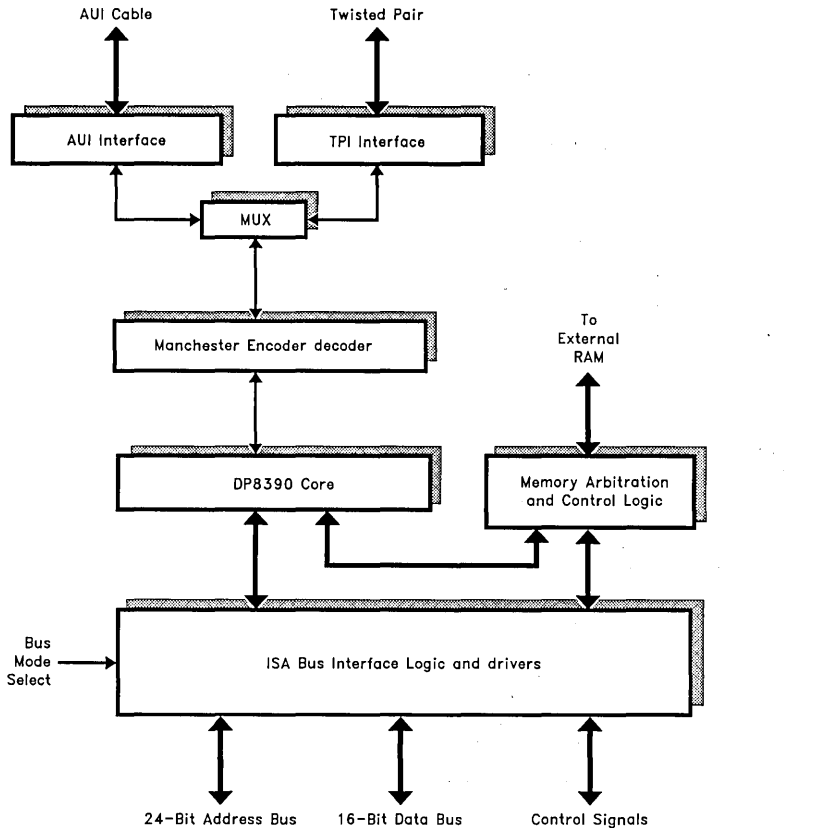


FIGURE 1. Block Diagram of DP83907

TL/F/12082-4

## 4.0 Functional Description

The DP83907 is a highly integrated and configurable Ethernet controller making it suitable for most Ethernet applications. The DP83907 integrates the functions of the following blocks:

1. ISA Bus Interface containing all logic required to connect the DP8390 core to a packet buffer RAM and the ISA bus.
2. DP8390 Ethernet Controller Core and Media Access Control logic.
3. Media Interface which includes a TPI transceiver.

### 4.1 BUS INTERFACE BLOCK

The DP83907's Bus interface block provides the circuitry to interface the Ethernet controller logic and the external packet buffer RAM to an ISA (Industry Standard Architecture)

Bus. The bus interface provides an NE2000 Architecture compatible I/O port architecture, supporting both 8-bit and 16-bit wide ISA Bus slots.

#### DETERMINING 8-BIT OR 16-BIT WIDE DATA

DP83907 can treat the system data bus and all internal data busses as 8 bits or 16 bits wide. 8-bit or 16-bit mode is determined by MSA9 at reset. For an adapter card this pin can be used to automatically detect if the card has been plugged into an 8-bit or 16-bit slot by connecting MSA9 via a 10 k $\Omega$  pull-up resistor to a V<sub>DD</sub> on the upper connector. It will be pulled high when plugged into a 16-bit slot, enabling 16-bit mode, and floating when plugged into an 8-bit slot. When floating the internal pull-down resistor will enable 8-bit mode.

## 4.0 Functional Description (Continued)

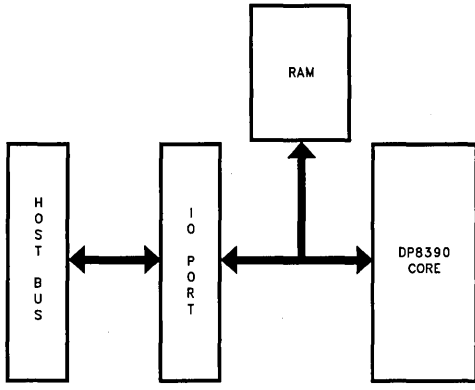


FIGURE 2. I/O Port

TL/F/12082-5

The DP83907's internal memory map is accessed one byte or word at a time, via a port within the systems I/O space. DP83907 is programmed by the user to control the transfers between its internal memory and the I/O port. The DP83907's internal registers and the memory access port are accessed within the system's I/O map. The address within this I/O map is, set by Configuration Register A.

### 16-BIT I/O PORT COMPATIBLE MODE I/O ADDRESS MAPPING

This mode is compatible with Novell's NE2000 Architecture. The base I/O address of DP83907 is configured by Configuration Register A (either upon power up or by software writing to this register). At the base I/O address the following structure appears.

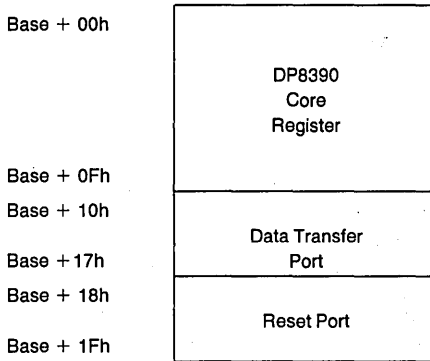


FIGURE 3. I/O Port Mode Register I/O Map

The DP83907 register space within this area are 8 bits wide, but the data transfer port is 16 bits wide. The DP83907's registers can be programmed to control the passing of data between its internal memory and the data transfer port. By accessing the data transfer port (using I/O instructions) the user can transfer data to or from the DP83907's internal memory. The DP83907's internal memory map is as shown below.

	D15	D0
0000h	00	PROM
001Fh	Aliased PROM	
4000h	8k x 16 Buffer RAM	
7FFFh	8k x 16 Buffer RAM	
8000h	Aliased PROM	
C000h	Aliased Buffer RAM	
FFFh	Aliased Buffer RAM	

(a)

	D15	D0
001Eh	00	57h
001Ch	00	57h
	•	•
	00	RESERVED
	•	•
00Ah	00	E'net Address 5
0008h	00	E'net Address 4
0006h	00	E'net Address 3
0004h	00	E'net Address 2
0002h	00	E'net Address 1
0000h	00	E'net Address 0

(b)

FIGURE 4. a) NIC Core's 16-Bit Memory Map  
b) 16-Bit PROM Map

DP83907 has a 64k address range but only does partial decoding through this space. The PROM data is mirrored from all decodes up to 4000H and the entire map is repeated at 8000H. To access either the PROM or the RAM the user must initiate a Remote DMA transfer between the I/O port and memory.

### 4.0 Functional Description (Continued)

On a remote read the DP83907 moves data from its internal memory map to the I/O port and the host system reads it by using an "INW" or "INSW" instruction from the I/O address of the data transfer port. If the system attempts to read the port before DP83907 has written the next word of data to it DP83907 will insert wait states into the system cycle using the CHRDY line. DP83907 will not begin the next memory read until the previous word of data has been read.

On a remote write the system writes data to the I/O port, using an "OUTW" or "OUTSW" instruction, and DP83907 moves it to its buffer memory. If the system attempts to write to the port before DP83907 has moved the data to memory, DP83907 will insert wait states into the system cycle using the CHRDY line. DP83907 will not begin the next memory write until a new word has been written to the I/O port.

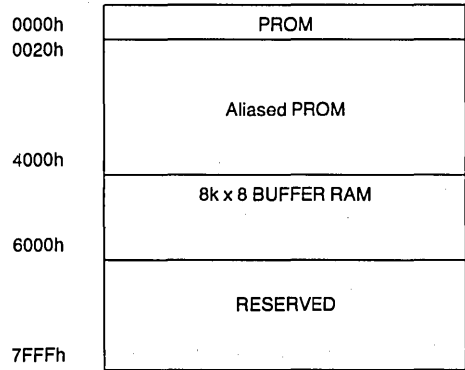
Addresses 00H to 1FH are specified as the PROM space for compatibility with the NE2000 Architecture. This is actually an array of 8-bit registers which are loaded from an external EEPROM after DP83907 is initialized by an ISA RESET. They should contain the same data as the PROM did in the NE2000 Architecture and in the same format. To transfer the data out the user must initiate a 16-bit DMA read transfer and discard the most significant byte of data on each transfer.

At address 00H of the PROM is a six byte Ethernet address for this node. The upper two addresses of the PROM store contain bytes which identify whether the DP83907 is in 8-bit or 16-bit mode. For 16-bit mode these bytes both contain the value 57H, for 8-bit mode they both contain 42H.

#### 8-BIT I/O PORT COMPATIBLE MODE

This mode is compatible with the 8-bit mode offered by Novells NE2000 Architecture. The NE2000 automatically detects whether it is in an 8-bit or 16-bit slot and configures itself appropriately. As explained in the previous paragraphs, the user can determine whether the board is in 8-bit or 16-bit mode by reading the PROM. In 8-bit mode only 8 Kbytes of RAM are addressable, as in the 8-bit mode of the NE2000 Architecture. The I/O map is the same as the 16-bit mode, the memory map is shown in *Figure 5*. Again the PROM has only a partial decode, so is mirrored at all addresses up to 4000H. The PROM still occupies 32 bytes of address space, although it only has 16 bytes of data, as the data at all odd address locations is merely a mirror of the data at the previous even address location.

A low cost card, using only one 8 Kbyte RAM, can be designed. If the function on MSA9 is left unconnected, then the DP83907 will always operate in 8-bit mode, regardless of the slot the board is in.



(a)

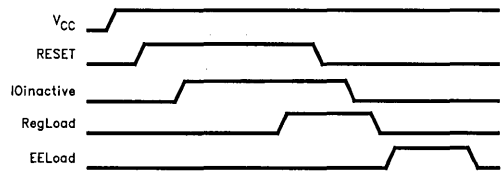
	D15	D0
1Eh	42h	42h
1Ch	42h	42h
	• RESERVED •	• RESERVED •
0Ah	E'net Address 5	E'net Address 5
08h	E'net Address 4	E'net Address 4
06h	E'net Address 3	E'net Address 3
04h	E'net Address 2	E'net Address 2
02h	E'net Address 1	E'net Address 1
00h	E'net Address 0	E'net Address 0

(b)

FIGURE 5. a) 8-Bit NIC Core's Memory Map  
b) 8-Bit PROM Map

#### 4.2 POWER ON RESET OPERATION

The DP83907 configures itself after a Reset signal is applied. To be recognized as a valid Power-On-Reset the Reset signal must be active for at least 415  $\mu$ s. *Figure 6* shows how the RESET circuitry operates.



TL/F/12082-6

FIGURE 6. RESET Operation

## 4.0 Functional Description (Continued)

The ISA standard determines that within 500 ns of RESET going active all devices should enter the appropriate reset condition. The DP83907 will generate the internal signal  $I_{Oinactive}$  after RESET has been active for 400 ns, which will disable all outputs and cause RESET to be the only input monitored. The DP83907 will not respond to a RESET pulse of shorter duration than this. An internal timer continues to monitor the amount of time RESET is active. After 415  $\mu$ s it is considered a valid Power-On-Reset and an internal signal called RegLoad is generated.

When a Power-On-Reset occurs the DP83907 latches in the values on the configuration pins and uses these to configure the internal registers and options. Internally these pins contain pull-down resistors, which are enabled when  $I_{Oinactive}$  goes active. If any pins are unconnected they default to a logic zero. The internal pull-down resistor has a high resistance to allow the external pull-up resistors to be of a high value. This limits the current taken by the memory support bus. The suggested external resistor value is 10 k $\Omega$ . The configuration registers are loaded from the memory support bus when RESET goes inactive if RegLoad is active. The internal pull-down resistors are enabled onto the bus until RegLoad has gone inactive.

A Power-On-Reset also causes the DP83907 to load the internal PROM space from the EEPROM, which can take up to 320  $\mu$ s. This occurs after RegLoad has gone inactive. The DP83907 will be inaccessible during this time. If EECONFIG is held high the configuration data loaded on the falling edge of RESET will be overwritten with data read from the serial EEPROM. Regardless of the level on EECONFIG, the PROM space will always be loaded with data from the serial EEPROM during the time specified as EELoad.

### 4.3 EEPROM OPERATION

The DP83907 uses an NMC93C06, or another serial EEPROM with compatible timings. The NMC93C06 is a 256-bit device, arranged as 16 words by 16 bits wide. The programmed contents of the EEPROM is shown in *Figure 7*.

	D15	D0
0Fh	EEPROM Code	Config. C
0Eh	Config. B	Config. A
	•	•
	•	•
	•	•
08h	42h	42h
07h	57h	57h
	•	•
	•	•
03h	Reserved	Reserved
02h	E'net Address 5	E'net Address 4
01h	E'net Address 3	E'net Address 2
00h	E'net Address 1	E'net Address 0

FIGURE 7. EEPROM Programming Map

### Mapping EEPROM into PROM Space

Data is read from the EEPROM at boot time and stored in registers within the DP83907. While this operation takes place the DP83907 can not be accessed by the system. These registers are mapped into the space traditionally occupied by the PROM in the NE2000

The user should program the EEPROM to contain the Ethernet address in the first six bytes and whatever is required in the next 8 bytes. The user should then program 5757H and 4242H into address 07h and 08h respectively. The DP83907 device driver may determine that this is a 16-bit board by checking this value.

The DP83907 reads the first 8 words from the EEPROM and maps them into the memory map at the appropriate address.

### In Circuit Programming the EEPROM

If the upper byte of address 0Fh in the EEPROM does not contain 073H then the DP83907 enters a mode that allows the EEPROM contents to be programmed. This can be used in production to program the EEPROM in-situ. By programming 073H into the uppermost byte the EEPROM is protected from future adaptation, except for configuration data which can always be modified.

If the EEPR bit of the Signature Register is low the EEPROM program mode may be entered. The EEPR bit is low if the EEPROM code is not programmed as 73H. In this mode, if the EELoad bit of Configuration Register B is set the user can directly control the EEPROM signals by writing to the Data Transfer Port. The user can write to the Data Transfer Port and the value on the SD3, SD2 and SD1 pins will be driven onto the EECS, SK and DI outputs. These outputs will be latched. The user can generate a clock on SK by repetitively writing 1 then 0 to the appropriate bit. This can be used to generate the EEPROM signals, as per the NM93C06 data sheet.

When the EEPROM has been programmed the user must give the DP83907 a reset signal to return to normal operation and to read in the new data.

### Storing and Loading Configuration from EEPROM

If the EECONFIG function on MSA10 is high during boot up the DP83907's configuration is read from the EEPROM, before the PROM data is read. The configuration data is stored within the upper two words of the EEPROM's address space. Configuration Registers A and B are located in the lower of these words, Register C in the lower byte of the upper word, as shown in *Figure 7*.

To write this configuration into the EEPROM the user must follow the routine specified in the pseudo code below. If the EEPROM code byte in the EEPROM is programmed as 73H the Configuration Registers may be written to in the EEPROM. This operation will work regardless of the level on EECONFIG. The EELoad bit of Configuration Register B being set starts the EEPROM write process. Care should be taken not to accidentally set the GDLink bit and therefore disable link integrity checking. The next 3 writes to this register load the values that will be stored in the configuration register (note that the last 2 of these writes do not have to follow the normal practice of preceding a write to this register with a read to this address). The DP83907 will then commence the EEPROM write. The write has been completed when the EELoad bit goes to zero. This loading process

## 4.0 Functional Description (Continued)

sure should be followed exactly and interrupts should be disabled until it has completed, to prevent any accidental accesses to the DP83907.

```
EEPROM_LOAD()
{
    DISABLE_INTERRUPTS();
    value = READ(CONFIG_B);
    value = value AND !GDLINK;
    value = value OR EELoad;
    WRITE(CONFIG_B, value);
    READ(CONFIG_B);
    WRITE(CONFIG_B, config_for_A);
    WRITE(CONFIG_B, config_for_B);
    WRITE(CONFIG_B, config_for_C);
    while (value AND EELoad)
    {
        value = READ(CONFIG_B);
        WAIT();
    }
    ENABLE_INTERRUPTS();
}
```

### 4.4 JUMPERLESS OPERATION SUPPORT

One of the biggest problems in installing new adapters in a PC is not knowing the available resources within that machine. DP83907's software configuration overcomes that problem. The conflicts possible in the I/O base selection can be overcome by a special mode for software configuration of the I/O base address. By using this mode, and by using the configuration storage capability of the EEPROM, a fully software configurable design on the ISA bus can be realized without address conflict problems.

This mode is invoked by having the DP83907 default to jumperless software configuration option in the I/O base selection. This mode enables configuration register A to be mapped to address location 278H which is defined to be a printer port's data register. If software writes to this location four consecutive times, on the fourth write the DP83907 will load the data written into the I/O address bits of Configuration Register A. This data should set the I/O base address to a known conflict-free value. The DP83907 can now be configured and operated at the desired base I/O address. If desired, the configuration software could change the EEPROM content to the new values eliminating the need to reconfigure upon each power up. Alternately the software could leave the EEPROM alone and execute the configuration using the printer port's data register upon each power up. This configuration scheme will only work once after each power-up. Therefore the user can not enable the DP83907 from reserved mode, change it back into reserved mode and enable it again. A power-on reset must occur between the first time it is enabled from the reserved mode and the second.

### 4.5 ETHERNET CABLE CONFIGURATION

DP83907 offers the choice of all the possible Ethernet cabling options, that is Ethernet (10BASE5), Thin Ethernet (10BASE2) and Twisted-pair Ethernet (10BASE-T). The

type of cabling used is controlled by Configuration Register B. DP83907 also supplies a THIN output signal which can be used to disable/enable an external DC-DC converter which is required for 10BASE2.

### 4.6 INTERRUPT AND LED OPERATION

The DP83907 has only one Interrupt Mode. There are 8 possible interrupts. Configuration Register A controls which of the 8 interrupt lines are driven, the others are TRI-STATE. **The interrupt outputs should be connected to the following ISA Interrupt lines, in the order given, to maintain NE2000 Architecture compatibility: 3, 4, 5, 9, 10, 11, 12, 15.**

### 4.7 BOOT PROM OPERATION

The DP83907 supports an optional boot PROM, the address and size of which can be set in Configuration Register C. This boot PROM can be any 8-bit wide storage device implemented with a non-volatile technology. Write cycles to this device can be enabled and disabled by programming Configuration Register B. This can be used to prevent unwanted write cycles to certain devices, such as a Flash EEPROM. The DP83907 supplies the chip select to the device and buffers the data on to and from the ISA bus, so the memory support data bus should be connected to the boot PROM's data pins.

### 4.8 DP8390 CORE (NETWORK INTERFACE CONTROLLER)

The DP8390 Core logic, *Figure 12*, contains the Serializer/Deserializer which is controlled by the Protocol PLA, DMA Control, FIFO, Address Comparator and Multicast Hashing Register. The DP8390 core implements all of the IEEE 802.3 Media access control functions for the DP83907 and interfaces to the internal ENDEC (on the left of the block diagram) and to the Bus Interface and memory support bus via a number of address, data and control signals (the right side of the block diagram). The following sections describe the functions of the DP8390 core.

#### Receive Deserializer

The Receive Deserializer is activated when the input signal Carrier Sense is asserted to allow incoming bits to be shifted into the shift register by the receive clock. The serial receive data is also routed to the CRC generator/checker. The Receive Deserializer includes a synch detector which detects the SFD (Start of Frame Delimiter) to establish where byte boundaries within the serial bit stream are located. After every eight receive clocks, the byte wide data is transferred to the 16-byte FIFO and the Receive Byte Count is incremented. The first six bytes after the SFD are checked for valid comparison by the Address Recognition Logic. If the Address Recognition Logic does not recognize the packet, the FIFO is cleared.

#### CRC Generator/Checker

During transmission, the CRC logic generates a local CRC field for the transmitted bit sequence. The CRC encodes all fields after the SFD. The CRC is shifted out MSB first follow-

## 4.0 Functional Description (Continued)

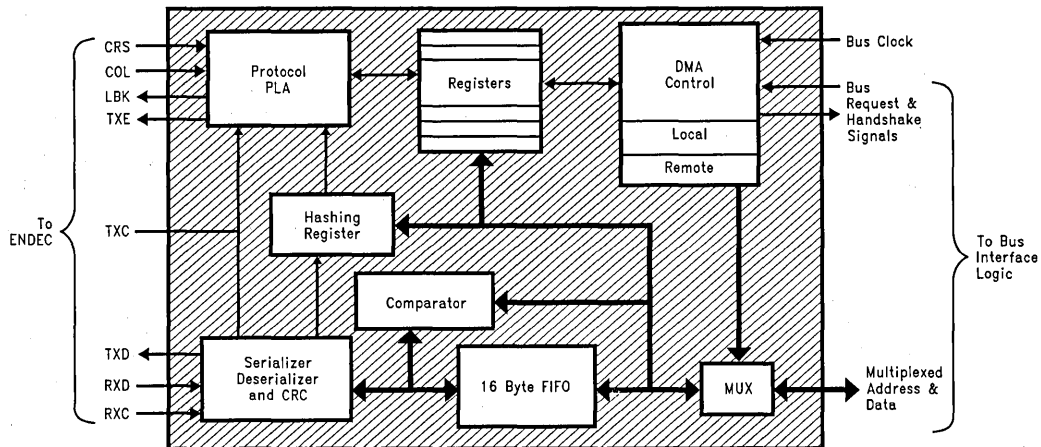


FIGURE 12. DP8390 Core Simplified Block Diagram

TL/F/12082-7

ing the last transmit byte. During reception the CRC logic generates a CRC field from the incoming packet. This local CRC is serially compared to the incoming CRC appended to the end of the packet by the transmitting node. If the local and received CRC match, a specific pattern will be generated and decoded to indicate no data errors. Transmission errors result in different pattern and are detected, resulting in rejection of a packet.

### Transmit Serializer

The Transmit Serializer reads parallel data from the FIFO and serializes it for transmission. The serializer is clocked by the transmit clock generated internally. The serial data is also shifted into the CRC generator/checker. At the beginning of each transmission, the Preamble and SFD Generator append 62 bits of 1,0 preamble and a 1,1 synch pattern. After the last data byte of the packet has been serialized the 32-bit FCS (Frame Check Sequence) field is shifted directly out of the CRC generator. In the event of a collision the Preamble and SFD generator is used to generate a 32-bit JAM pattern of all 1's.

### Comparator-Address Recognition Logic

The address recognition logic compares the Destination Address Field (first 6 bytes of the received packet) to the Physical address registers stored in the Address Register Array.

If any one of the six bytes does not match the pre-programmed physical address, the Protocol Control Logic rejects the packet. All multicast destination addresses are filtered using a hashing technique. (See register description.) If the multicast address indexes a bit that has been set in the filter bit array of the Multicast Address Register Array the packet is accepted, otherwise it is rejected by the Protocol Control Logic. Each destination address is also checked for all 1's which is the reserved broadcast address.

### FIFO and Packet Data Operations

#### OVERVIEW

To accommodate the different rates at which data comes from (or goes to) the network and goes to (or comes from) the packet buffer memory, the DP83907 contains a 16-byte FIFO for buffering data between the media and the buffer RAM located on the memory support bus. The FIFO threshold is programmable. When the FIFO has filled to its programmed threshold, the local DMA channel transfers these bytes (or words) into local memory (via the memory bus). It is crucial that the local DMA is given access to the bus within a minimum bus latency time; otherwise a FIFO under-run (or overrun) occurs.

## 4.0 Functional Description (Continued)

FIFO underruns or overruns are caused when a local DMA request is issued while an ISA bus access is current and the ISA cycle takes longer to complete than the local DMA's tolerable latency. This tolerable latency depends on the FIFO threshold, whether it is in byte or word wide mode and the speed of the DMA clock (BSCLK frequency). Note that this refers to standard ISA cycles, NOT those where the CHRDY is deasserted extending the cycle.

### FIFO THRESHOLD DETECTION

To assure that no overwriting of data in the FIFO, the FIFO logic flags a FIFO overrun as the 13th byte is written into the FIFO, effectively shortening the FIFO to 13 bytes. The FIFO logic also operates differently in Byte Mode and in Word Mode. In Byte Mode, a threshold is indicated when the  $n + 1$  byte has entered the FIFO; thus, with an 8 byte threshold, the DP83907 issues a request to the buffer RAM when the 9th byte has entered the FIFO, making the effective threshold 9 bytes. For Word Mode, the request is not generated until the  $n + 2$  bytes have entered the FIFO. Thus, with a 4 word threshold (equivalent to 8 byte threshold), a request to the buffer RAM is issued when the 10th byte has entered the FIFO, making the effective threshold 10 bytes.

### TOLERABLE LATENCY CALCULATION

To prevent a FIFO **overrun** a byte (or word) of data must be **removed** from the FIFO before the 13th byte is written. Therefore the worst case tolerable latency is the time from the effective threshold being reached to the time the 13th byte is written minus the time taken to load the first byte (or word) of data to the FIFO during a local DMA burst (8 BSCLKs).

$$\begin{aligned} \text{tolerable latency} &= ((\text{overrun} - \text{effective}) \text{ threshold} \\ &\quad \times \text{time to transfer byte on network}) \\ &\quad - \text{time to fill 1st FIFO location} \end{aligned}$$

For the case of a 4 word threshold using a 20 MHz BSCLK:

$$\begin{aligned} \text{tolerable latency} &= ((13 - 10) \times 800) - (8 \times 50) \text{ ns} \\ &= 2 \mu\text{s} \end{aligned}$$

To prevent a FIFO **underrun** a byte (or word) of data must be **added** to the FIFO before the last byte is removed. Therefore the worst case tolerable latency is the time from the effective threshold being reached to the time the last byte is removed minus the time taken to load the first byte (or word) of data to the FIFO during a local DMA burst (8 BSCLKs).

$$\begin{aligned} \text{tolerable latency} &= (\text{threshold} \\ &\quad \times \text{time to transfer byte on network}) \\ &\quad - \text{time to fill 1st FIFO location} \end{aligned}$$

For the case of a 4 word threshold using a 20 MHz BSCLK:

$$\begin{aligned} \text{tolerable latency} &= (4 \times 800) - (8 \times 50) \text{ ns} \\ &= 2.8 \mu\text{s} \end{aligned}$$

The worst case latency, either overrun or underrun, ultimately limits the overall latency that the DP83907 can tolerate. If the standard ISA cycles are shorter than the worst case latency, then no FIFO overruns or underruns will occur.

### BEGINNING OF RECEIVE

At the beginning or reception, the DP83907 stores the entire Address field of each incoming packet in the FIFO to deter-

mine whether the packet matches its Physical Address Registers or maps to one of its Multicast Registers. This causes the FIFO to accumulate 8 bytes. Furthermore, there are some synchronization delays in the DMA PLA. Thus, the actual time that a request to access the buffer RAM is asserted from the time the Start of Frame Delimiter (SFD) is detected is 7.8  $\mu\text{s}$ . This operation affects the bus latencies at 2 and 4 byte thresholds during the first receive request since the FIFO must be filled to 8 bytes (or 4 words) before issuing a request to the buffer RAM.

### END OF RECEIVE

When the end of a packet is detected by the ENDEC module, the DP83907 enters its end of packet processing sequence, emptying its FIFO and writing the status information at the beginning of the 1st buffer. The DP83907 holds onto the memory bus for the entire sequence. The longest time that local DMA will hold the buffer RAM occurs when a packet ends just as the DP83907 performs its last FIFO burst. The DP83907, in this case, performs a programmed burst transfer followed by flushing the remaining bytes in the FIFO, and completed by writing the header information to the buffer memory. The following steps occur during this sequence.

1. DP83907 issues request to access the RAM because the FIFO threshold has been reached.
2. During the burst, the packet ends, resulting in the request being extended.
3. DP83907 flushes remaining bytes from FIFO.
4. DP83907 performs internal processing to prepare for writing the header.
5. DP83907 writes 4-byte (2-word) header
6. DP83907 de-asserts access to the buffer RAM.

### BEGINNING OF TRANSMIT

Before transmitting, the DP83907 performs a prefetch from memory to load the FIFO. The number of bytes prefetched is the programmed FIFO threshold. The next request to the buffer RAM is not issued until after the DP83907 actually begins transmitting data, i.e., after SFD.

### READING THE FIFO

If the FIFO is read during normal operation the DP83907 will "hang" the ISA bus by deasserting CHRDY and never asserting it. The FIFO should only be read during loopback diagnostics.

### PROTOCOL PLA

The protocol PLA is responsible for implementing the IEEE 802.3 protocol, including collision recovery with random backoff. The Protocol PLA also formats packets during transmission and strips preamble and synch during reception.

### DMA AND BUFFER CONTROL LOGIC

The DMA and Buffer Control Logic is used to control two 16-bit DMA channels. During reception, the Local DMA stores packets in a receive buffer ring, located in buffer memory. During transmission the Local DMA uses programmed pointer and length registers to transfer a packet from local buffer memory to the FIFO.

## 4.0 Functional Description (Continued)

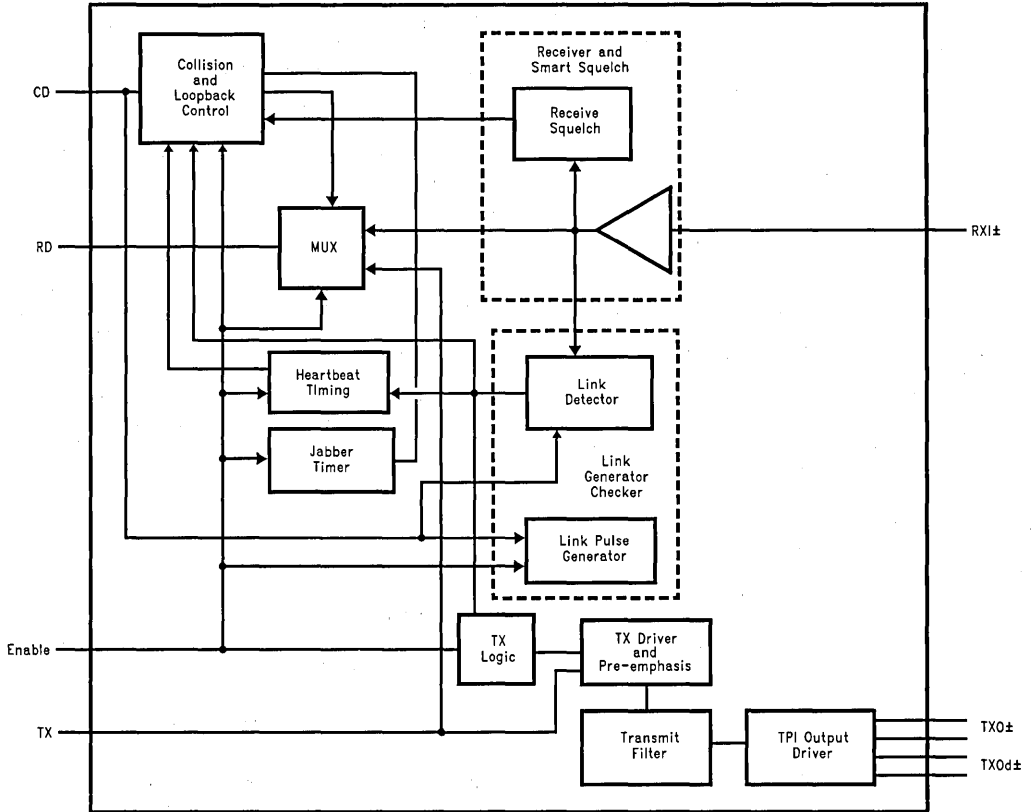


FIGURE 13. Twisted Pair Interface Module Block Diagram

TL/F/12082-8

A second DMA channel (Remote DMA) is used as a slave DMA to transfer data between the local buffer memory and the host system. The Local DMA and Remote DMA are internally arbitrated, with the Local DMA channel having highest priority. Both DMA channels use a common external bus clock to generate all required bus timing. External arbitration is performed with a standard bus request, bus acknowledge handshake protocol.

### 4.9 TWISTED PAIR INTERFACE MODULE

The TPI consists of five main logical functions:

- The Receiver/Smart Squelch, responsible for determining when valid data is present on the differential receive inputs (RXI $\pm$ ) and receiving the data.
- The Collision function checks for simultaneous transmission and reception of data on the TXO $\pm$  and RXI $\pm$  pins.
- The Link Detector/Generator checks the integrity of the cable connecting the two twisted pair MAUs.
- The Jabber disables the transmitter if it attempts to transmit a longer than legal packet.
- The TX Driver and Pre-emphasis transmits Manchester encoded data to the twisted pair network via the summing resistors and transformer/filter.



## 4.0 Functional Description (Continued)

### Receiver and Smart Squelch

The DP83907 implements an intelligent receive squelch on the RX1± differential inputs to ensure that impulse noise on the receive inputs will not be mistaken for a valid signal.

The squelch circuitry employs a combination of amplitude and timing measurements to determine the validity of data on the twisted pair inputs. There are two voltage level options for the smart squelch. One mode, 10BASE-T mode, uses levels that meet the 10BASE-T specification. The second mode, reduced squelch mode, uses a lower squelch threshold level, and can be used in longer cable applications where due to attenuation smaller signal levels may be present. The squelch level mode can be selected in the DP83907 configuration registers.

Figure 14 shows the operation of the smart squelch in 10BASE-T mode.

The signal at the start of packet is checked by the smart squelch and any pulses not exceeding the squelch level (either positive or negative, depending upon polarity) will be rejected. Once this first squelch level is overcome correctly the opposite squelch level must then be exceeded within 150 ns later. Finally the signal must exceed the original squelch level within a further 150 ns to ensure that the input waveform will not be rejected. The checking procedure results in the loss of typically three bits at the beginning of each packet.

Only after all these conditions have been satisfied will a control signal be generated to indicate to the remainder of the circuitry that valid data is present. At this time the smart squelch circuitry is reset.

In the reduced squelch mode the operation is identical except that the lower squelch levels shown in the figure are used.

Valid data is considered to be present until either squelch level has not been generated for a time longer than 150 ns, indicating End of Packet. Once good data has been detected the squelch levels are reduced to minimize the effect of noise causing premature End of Packet detection.

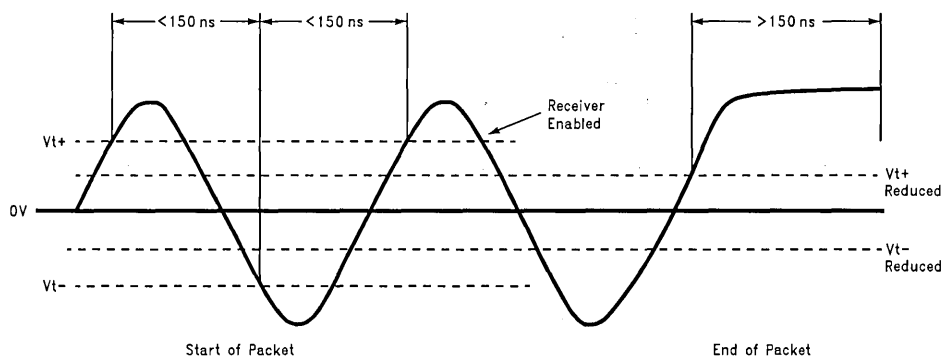


FIGURE 14. Twisted Pair Squelch Waveform

TL/F/12082-9

### Collision

A collision is detected by the TPI module when the receive and transmit channels are active simultaneously. If the TPI is receiving when a collision is detected it is reported to the controller immediately. If, however, the TPI is transmitting when a collision is detected the collision is not reported until seven bits have been received while in the collision state. This prevents a collision being reported incorrectly due to noise on the network. The signal to the controller remains for the duration of the collision.

Approximately 1  $\mu$ sec after the transmission of each packet a signal called the Signal Quality Error (SQE) consisting of typically 10 cycles of 10 MHz is generated. This 10 MHz signal, also called the Heartbeat, ensures the continued functioning of the collision circuitry.

### Link Detector/Generator

The link generator is a timer circuit that generates a link pulse as defined by the 10 Base-T specification that will be generated by the transmitter section. The pulse which is 100 ns wide is transmitted on the TXO+ output, every 16 ms, in the absence of transmit data.

The pulse is used to check the integrity of the connection to the remote MAU. The link detection circuit checks for valid pulses from the remote MAU and if valid link pulses are not received the link detector will disable the transmit, receive and collision detection functions.

The GDLNK output can directly drive a LED to show that there is a good twisted pair link. For normal conditions the LED will be on. The link integrity function can be disabled by setting the GDLNK bit of Configuration Register B.

### Jabber

The jabber timer monitors the transmitter and disables the transmission if the transmitter is active for greater than 26 ms. The transmitter is then disabled for the whole time that the Endec module's internal transmit enable is asserted. This signal has to be deasserted for approximately 750 ms (the unjab time) before the Jabber re-enables the transmit outputs.

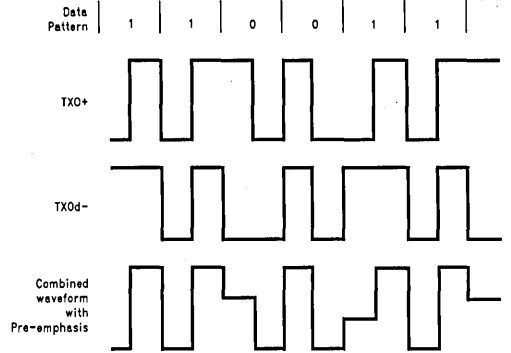
## 4.0 Functional Description (Continued)

### Transmitter

The transmitter consists of four signals, the true and complement Manchester encoded data (TXO±) and these signals delayed by 50 ns (TXOd±).

These four signals are resistively combined TXO+ with TXOd- and TXO- with TXOd+. This is known as digital pre-emphasis and is required to compensate for the twisted pair cable which acts like a low pass filter causing greater attenuation to the 10 MHz (50 ns) pulses of the Manchester encoded waveform than the 5 MHz (100 ns) pulses.

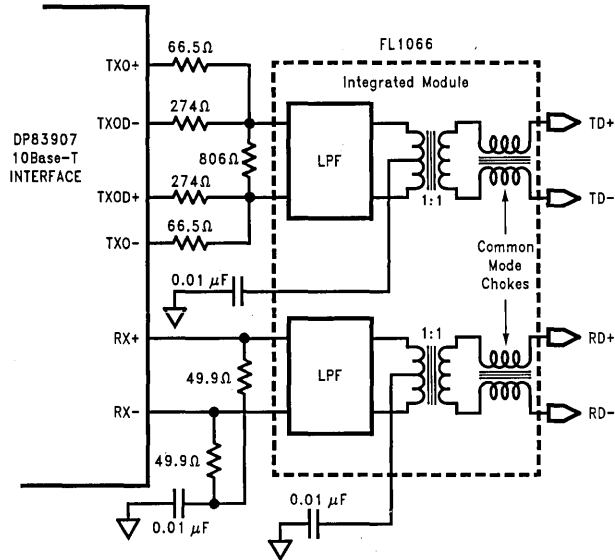
An example of how these signals are combined is shown in the following diagram.



TL/F/12082-10

FIGURE 15. Typical Summed Transmit Waveform

The signal with pre-emphasis shown above is generated by resistively combining TXO+ and TXOd-. This signal along with its complement is passed to the transmit filter.



TL/F/12082-11

FIGURE 16a. Circuitry to Connect DP83907 to Twisted Pair Cable with External Filter

## 4.0 Functional Description (Continued)

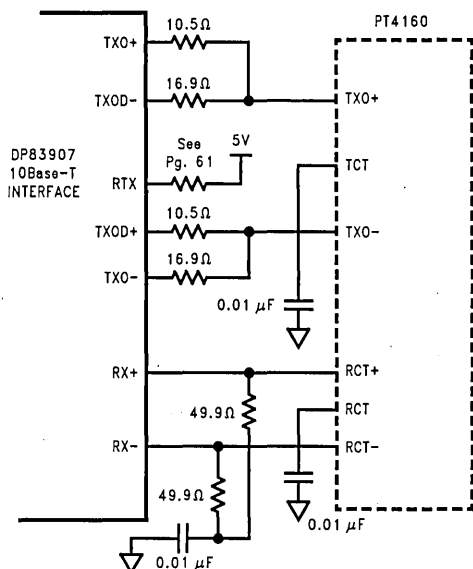


FIGURE 16b. Circuitry to Connect DP83907 to Twisted Pair Cable with Internal Filter

TL/F/12082-52

### On-Chip Filters

The on-chip filters are enabled via an external pull-up resistors on MSD<12> at configuration. Only an isolation transformer and impedance matching resistors are needed for the transmit and receive twisted pair interface.

### UTP/STP Function

The TPI transceiver supports both shielded and unshielded twisted pair cable. UTP is default but STP can be enabled during configuration by a pull-up on MSA<7> or by setting bit D6 of configuration register C. In UTP mode TXO+ and TXOD+ are driven and TXO- and TXOD- are TRI-STAT-ED. In STP mode, TXO- and TXOD- are driven and TXO+ and TXOD+ are TRI-STAT-ED.

### Auto-Switch Function

When an auto-switch function is enabled at configuration by a pull-up on MSA<5> or by setting bit D4 of the configuration register C, it allows the transceiver to switch between TP and AUI outputs. If there is an absence of link pulses, the transceiver will switch to AUI mode. Similarly, when the transceiver starts detecting link pulses it will switch to TP mode. The switching from one mode to the next is only done after the current package has been transmitted or received. If the twisted pair output is jabbering and gets into link fail state, then the switch to AUI mode is only done after the jabbering is done, including the time it takes to unjab (unjab time). When auto-switching is enabled the THIN output is automatically generated if AUI is selected.

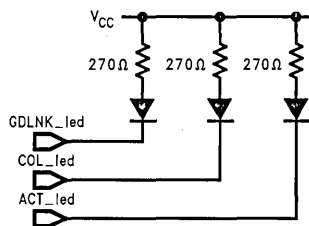


FIGURE 17. Typical DP83907 LED Connection

TL/F/12082-12

### Status Information

Status information is provided by the DP83907 on the ACT\_led, COL\_led and GDLNK\_led outputs as described in the pin description table. These outputs are suitable for driving status LED's as shown in Figure 17. All outputs are open drain.

Recommended Transformers for Internal Filter mode:

- 1) Valor PT4160
- 2) Pulse PE-68029
- 3) PCA EPE6087A
- 4) Delta THX16B02
- 5) Belfus A553-3899-06
- 6) Kappa TP3036



## 4.0 Functional Description (Continued)

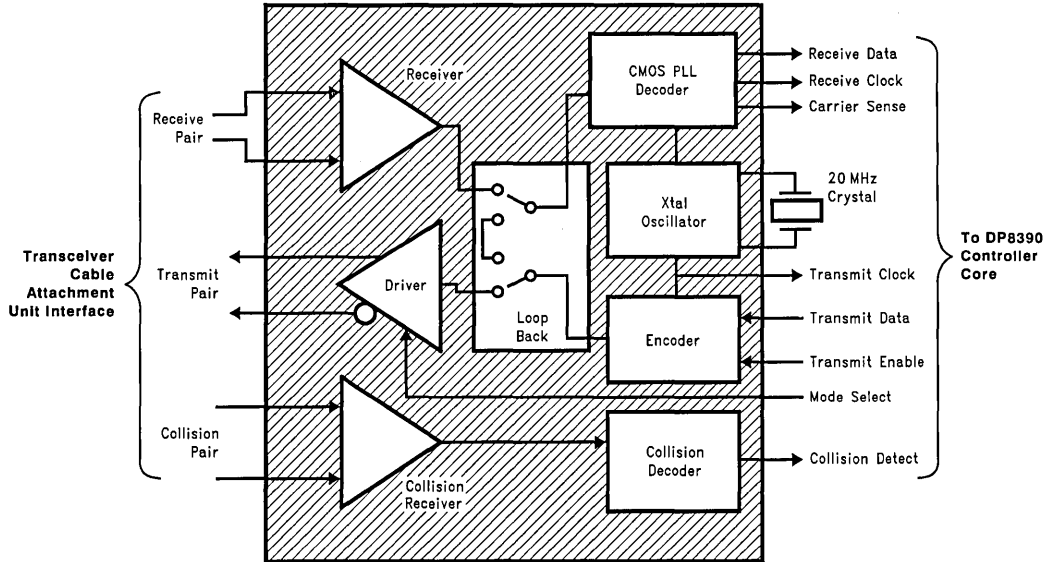


FIGURE 18. Encoder/Decode Block Diagram

TL/F/12082-13

### 4.10 ENCODER/DECODER (ENDEC) MODULE

The ENDEC consists of four main logical blocks:

- The oscillator generates the 10 MHz transmit clock signal for system timing.
- The Manchester encoder accepts NRZ data from the controller, encodes the data to Manchester, and transmits the data differentially to the transceiver, through the differential transmit driver.
- The Manchester decoder receives Manchester data from the transceiver, converts it to NRZ data and clock pulses, and sends it to the controller.
- The collision translator indicates to the controller the presence of a valid 10 MHz collision signal to the PLL.

#### Oscillator

The oscillator is controlled by a 20 MHz parallel resonant crystal connected between X1 and X2 or by an external clock on X1. The 20 MHz output of the oscillator is divided by 2 to generate the 10 MHz transmit clock for the controller. The oscillator also provides internal clock signals to the encoding and decoding circuits.

#### CRYSTAL OPERATION

If the crystal used with the internal oscillator circuit is not properly selected, the DP83907 oscillator may not reliably start oscillation under all conditions.

If this occurs, it could be deceiving to a designer, since his prototypes may work fine. However, when the designer does qualification testing or starts production, he may en-

counter a higher than expected board yield loss due to the oscillator not starting. The DP83907's oscillator circuit clocks the Encoder-Decoder logic. The DP83907's oscillator also clocks the twisted pair interface block. If the oscillator does not start, the DP83907 will not be able to transmit or receive.

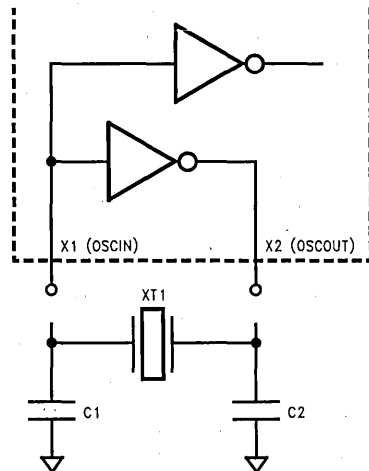


FIGURE 19. Crystal Connection to DP83907 (see text for component values)

TL/F/12082-14

## 4.0 Functional Description (Continued)

If a crystal is connected to the DP83907, it is recommended that the circuit shown in *Figure 19* be used and that the components used meet the following:

- Crystal XT1: AT cut parallel resonant crystal
- Series Resistance:  $\leq 25\Omega$
- Specified Load Capacitance:  $\leq 20$  pF
- Accuracy: 0.005% (50 ppm)
- Typical Load:  $50 \mu W - 75 \mu W$

The recommended values for capacitors C1 and C2 should be 26 pF minus the board capacitance on that pin. Therefore if both X1 and X2 have 4 pF of board capacitance then a 22 pF capacitor should be used.

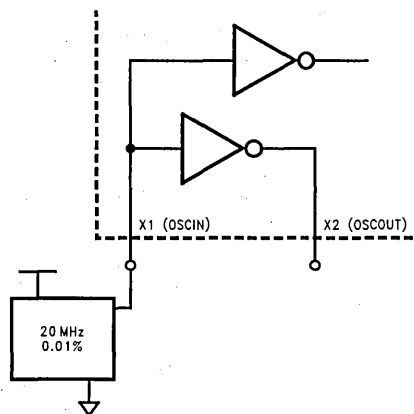
According to the IEEE 802.3 standard, the entire oscillator circuit (crystal and amplifier) must be accurate to 0.01%. When using a crystal, the X2 pin is not guaranteed to provide a TTL compatible logic output, and should not be used to drive external standard logic. If additional logic needs to be driven, then an external oscillator should be used, as described in the following section.

### Oscillator Module Operation

If the designer wishes to use a crystal clock oscillator, one that provides the following should be employed:

- 1) TTL or CMOS output with a 0.01% frequency tolerance
- 2) 40–60% duty cycle

The circuit is shown in *Figure 20*. When using a clock oscillator it is recommended that the designer connect the oscillator output to the X1 pin and leave the X2 pin floating.

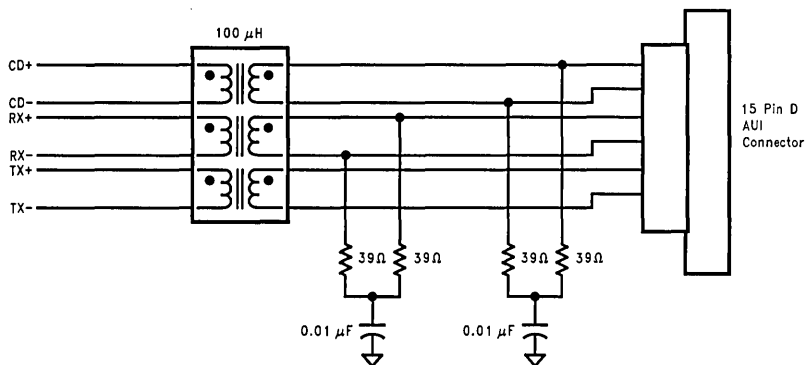


TL/F/12082-15

**FIGURE 20. DP83907 Connection for Oscillator Module Manchester Encoder and Differential Driver**

The differential transmit pair, on the secondary of the employed transformer, drives up to 50 meters of twisted pair AUI cable.

The DP83907 allows full-step to be compatible with IEEE 802.3. Transmit+ and Transmit- are equal in the idle state, providing zero differential voltage to operate with transformer coupled loads.



TL/F/12082-16

**FIGURE 21. Connection from DP83907's AUI Port to a AUI Connector**

## 4.0 Functional Description (Continued)

### Manchester Decoder

The decoder consists of a differential receiver and a PLL to separate a Manchester encoded data stream into internal clock signals and data. The differential input must be externally terminated with two  $39\Omega$  resistors connected in series if the standard  $78\Omega$  transceiver drop cable is used. In thin Ethernet applications, these resistors are optional. To prevent noise from falsely triggering the decoder, a squelch circuit at the input rejects signals with levels less than  $-175$  mV. Signals more negative than  $-300$  mV. Data becomes valid typically within 6-bit times. The DP83907 may tolerate bit jitter up to 20 ns in the received data. The decoder detects the end of a frame when no more mid-bit transitions are detected.

### Collision Translator

When in AUI Mode, the Ethernet transceiver (DP8392 CTI) detects a collision, it generates a 10 MHz signal to the differential collision inputs ( $CD\pm$ ) of the DP83907. When these inputs are detected active, the DP83907 uses this signal to back off its current transmission and reschedule another one.

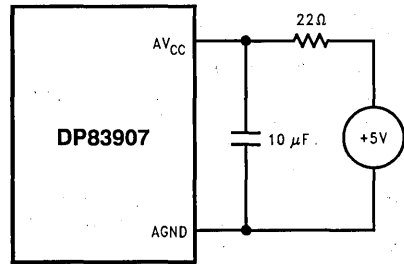
In this mode the  $COL_{led}$  output will indicate when the  $CD+$  lines are active during activity on the network. This means it will correctly indicate any collision on the network, but will not be lit for heartbeat or if there is no cable connected.

The collision differential inputs are terminated the same way as the differential receive inputs. The squelch circuitry is also similar, rejecting pulses levels less than  $-175$  mV.

### AV<sub>CC</sub> Power Supply Consideration

The AV<sub>CC</sub> pin is the +5V power supply for the phase lock loop (PLL) of the ENDEC unit. Since this is an analog circuit, excessive noise on the AV<sub>CC</sub> pin can affect the performance of the PLL. This noise, if in the 10 KHz–400 KHz range, can reduce the jitter performance of the ENDEC, resulting in missing packets or CRC errors.

If the power supply noise is causing significant packet reception error, a low pass filter could be added to reduce the power supply noise and hence improve the jitter performance. Standard analog design techniques should be utilized when laying out the power supply traces on the board. If the digital power supply is used, it may be desirable to add a one pole RC filter (designed to have a cut-off frequency of 1 KHz) as shown in Figure 22 to improve the jitter performance. The AV<sub>CC</sub> draws 3 mA–4 mA so the voltage across the resistor is less than 90 mV, which will not affect the PLL's operation.



TL/F/12082-17

FIGURE 22. Filtering Power Supply Noise

## 5.0 Register Descriptions

### 5.1 CONFIGURATION REGISTERS

These registers are used to configure the operation of the DP83907 typically after power up. These registers control the configuration of bus interface, setting options like interrupt selection, I/O base address, and other specific modes.

#### MODE CONFIGURATION REGISTER A

To prevent any accidental writes of this register it is "hidden" behind a previously unused register. Register 0AH in the DP83907's Page 0 of registers was previously reserved on a read. Now Configuration Register A can be read at that address and can be written to by following a read to 0AH with a write to 0AH. If any other DP83907 register accesses take place between the read and the write then the write to 0AH will access the Remote Byte Count Register 0.

7	6	5	4	3	2	1	0
RES	FRd/Wr	INT2	INT1	INT0	IOAD2	IOAD1	IOAD0

Bit	Symbol	Function																
0-2	IOAD0-IOAD2	<p><b>I/O ADDRESS:</b> These three bits determine the base I/O address of the DP83907 within the system's I/O map. The DP83907 occupies 20H bytes of the system's address space.</p> <p>Bit 2 1 0</p> <table style="margin-left: 20px;"> <tr><td>0 0 0</td><td>0300H</td></tr> <tr><td>0 0 1</td><td>Software (Note 1)</td></tr> <tr><td>0 1 0</td><td>0240H</td></tr> <tr><td>0 1 1</td><td>0280H</td></tr> <tr><td>1 0 0</td><td>02C0H</td></tr> <tr><td>1 0 1</td><td>0320H</td></tr> <tr><td>1 1 0</td><td>0340H</td></tr> <tr><td>1 1 1</td><td>0360H</td></tr> </table> <p><b>Note 1:</b> When 001 is selected the DP83907 will not respond to any I/O Addresses, but will allow 4 consecutive writes to 278H to write these three bits of this register. This sequence will only operate once after a power-on reset. This mode allows the DP83907 to be configured via software without conflicting with other peripherals.</p>	0 0 0	0300H	0 0 1	Software (Note 1)	0 1 0	0240H	0 1 1	0280H	1 0 0	02C0H	1 0 1	0320H	1 1 0	0340H	1 1 1	0360H
0 0 0	0300H																	
0 0 1	Software (Note 1)																	
0 1 0	0240H																	
0 1 1	0280H																	
1 0 0	02C0H																	
1 0 1	0320H																	
1 1 0	0340H																	
1 1 1	0360H																	
3-5	INT0-INT2	<p><b>INTERRUPT LINE USED:</b></p> <p>Bit 5 4 3    IRQ</p> <table style="margin-left: 20px;"> <tr><td>0 0 0</td><td>3</td></tr> <tr><td>0 0 1</td><td>4</td></tr> <tr><td>0 1 0</td><td>5</td></tr> <tr><td>0 1 1</td><td>9</td></tr> <tr><td>1 0 0</td><td>10</td></tr> <tr><td>1 0 1</td><td>11</td></tr> <tr><td>1 1 0</td><td>12</td></tr> <tr><td>1 1 1</td><td>15</td></tr> </table>	0 0 0	3	0 0 1	4	0 1 0	5	0 1 1	9	1 0 0	10	1 0 1	11	1 1 0	12	1 1 1	15
0 0 0	3																	
0 0 1	4																	
0 1 0	5																	
0 1 1	9																	
1 0 0	10																	
1 0 1	11																	
1 1 0	12																	
1 1 1	15																	
6	FRd/WR	<p><b>FAST RD/WR:</b> When this bit is set high the DP83907, in I/O mode, will begin the next port fetch before the current IORD/IOWR has completed. In slow ISA systems this may cause the data in the port to be overwritten before the ISA cycle has been completed.</p>																
7	RES	<p><b>RESERVED:</b> This bit must be set low for normal operation.</p>																

## 5.0 Register Descriptions (Continued)

### MODE CONFIGURATION REGISTER B

To prevent any accidental writes of this register it is "hidden" behind a previously unused register. Register 0BH in the DP83907's Page 0 of registers was previously reserved on a read. Now Configuration Register B can be read at that address and can be written to by following a read to 0BH with a write to 0BH. If any other DP83907 register accesses take place between the read and the write then the write to 0BH will access the Remote Byte Count Register 1. **Care should be taken when writing to this register as GDLINK and BE are not simple read/write bits.** e.g., the user can not change the physical layer by reading B, or-ing the returned value with the bits to be set and writing this value to B. This could inadvertently disable link integrity generation and clear a bus error indication before it was noted.

7	6	5	4	3	2	1	0
EELOAD	BPWR	BE	RES	IO16CON	GDLINK	PHYS1	PHYS0

Bit	Symbol	Function
0-1	PHYS0- PHYS1	<p><b>PHYSICAL LAYER INTERFACE:</b> These 2 bits determine which type of physical interface the DP83907 is using. The 2 TPI interfaces use twisted pair outputs and inputs, while the other 2 interfaces use the AUI outputs and inputs. In 10BASE5 mode the THICK/THIN output pin is driven low, in 10BASE2 mode it is driven high. This can be used to enable the DC-DC converter required by the 10BASE2 specification to provide electrical isolation. The Non spec TPI mode is a twisted pair mode with reduced receive squelch levels. This allows the use of longer cable lengths than specified in the twisted pair specification, or the use of cable with higher losses.</p> <p>Bit 1 0</p> <p>0 0 TPI (10BASE-T Compatible Squelch Level)</p> <p>0 1 Thin Ethernet (10BASE2)</p> <p>1 0 Thick Ethernet (10BASE5) (AUI Port)</p> <p>1 1 TPI (Reduced Squelch Level)</p>
2	GDLINK	<p><b>GOOD LINK:</b> When a 1 is written to this bit the link test pulse generation and integrity checking is disabled. When this bit is read it will indicate the link status, reflecting the value shown on the LED output. It is 0 if the DP83907 is in AUI mode or if link testing is enabled and the link integrity is bad (i.e., the twisted pair link has been broken). It is 1 if the DP83907 is in TPI mode, link integrity checking is enabled and the link integrity is good (i.e., the twisted pair link has <b>not</b> been broken) or if the link testing is disabled.</p>
3	IO16CON	<p><b>IO16 CONTROL:</b> When this bit is set high the DP83907 generates <math>\overline{IO16}</math> after <math>\overline{IOR\overline{D}}</math> or <math>\overline{IOW\overline{R}}</math> go active. If low this output is generated only on address decode.</p>
4	RES	<p><b>RESERVED:</b> This bit must be set low for normal operation.</p>
5	BE	<p><b>BUS ERROR:</b> This bit shows that the DP83907 has detected a bus error condition. This will go high if the DP83907 attempts to insert wait states into a system access and the system terminates the cycle without inserting the wait states. Writing a one to this bit clears it to zero. Writing a zero has no effect.</p>
6	BPWR	<p><b>BOOT PROM WRITE:</b> When this bit is low no write cycles are generated to the boot PROM.</p>
7	EELOAD	<p><b>EEPROM LOAD:</b> Writing a 1 to this bit enables the EEPROM load algorithm as detailed in Section 4. <b>This bit should not be configured to be high, either from switches or an EEPROM.</b></p>



## 5.0 Register Descriptions (Continued)

### HARDWARE CONFIGURATION REGISTER C

This register is configured during a RESET and can be accessed by 3rd consecutive read config register A.

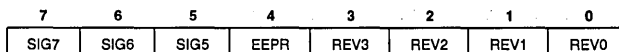
7	6	5	4	3	2	1	0
SOFEN	RES	RES	RES	BPS3	BPS2	BPS1	BPS0

Bit	Symbol	Function																																																																																																
0-3	BPS0-3	<p><b>BOOT PROM SELECT:</b> Selects address at which boot PROM begins and the size. When the system reads within the selected memory area DP83907 reads the data in through MSD0-7 and drives it onto the system data bus. The following are valid addresses and sizes:</p> <table border="1"> <thead> <tr> <th>Bit 3</th> <th>Bit 2</th> <th>Bit 1</th> <th>Bit 0</th> <th>Address</th> <th>Size</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>X</td> <td>X</td> <td>No Boot PROM</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>0</td> <td>0C000H</td> <td>8K</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td>0C400H</td> <td>8K</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>0</td> <td>0C800H</td> <td>8K</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>1</td> <td>0CC00H</td> <td>8K</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>0</td> <td>0D000H</td> <td>8K</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>1</td> <td>0D400H</td> <td>8K</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>0</td> <td>0D800H</td> <td>8K</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>1</td> <td>0DC00H</td> <td>8K</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>0</td> <td>0C000H</td> <td>32K</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>1</td> <td>0C800H</td> <td>32K</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>0</td> <td>0D000H</td> <td>32K</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>1</td> <td>0D800H</td> <td>32K</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>0C000H</td> <td>64K</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>0D000H</td> <td>64K</td> </tr> </tbody> </table>	Bit 3	Bit 2	Bit 1	Bit 0	Address	Size	0	0	0	X	X	No Boot PROM	0	0	1	0	0C000H	8K	0	0	1	1	0C400H	8K	0	1	0	0	0C800H	8K	0	1	0	1	0CC00H	8K	0	1	1	0	0D000H	8K	0	1	1	1	0D400H	8K	1	0	0	0	0D800H	8K	1	0	0	1	0DC00H	8K	1	0	1	0	0C000H	32K	1	0	1	1	0C800H	32K	1	1	0	0	0D000H	32K	1	1	0	1	0D800H	32K	1	1	1	0	0C000H	64K	1	1	1	1	0D000H	64K
Bit 3	Bit 2	Bit 1	Bit 0	Address	Size																																																																																													
0	0	0	X	X	No Boot PROM																																																																																													
0	0	1	0	0C000H	8K																																																																																													
0	0	1	1	0C400H	8K																																																																																													
0	1	0	0	0C800H	8K																																																																																													
0	1	0	1	0CC00H	8K																																																																																													
0	1	1	0	0D000H	8K																																																																																													
0	1	1	1	0D400H	8K																																																																																													
1	0	0	0	0D800H	8K																																																																																													
1	0	0	1	0DC00H	8K																																																																																													
1	0	1	0	0C000H	32K																																																																																													
1	0	1	1	0C800H	32K																																																																																													
1	1	0	0	0D000H	32K																																																																																													
1	1	0	1	0D800H	32K																																																																																													
1	1	1	0	0C000H	64K																																																																																													
1	1	1	1	0D000H	64K																																																																																													
4	RES	<b>RESERVED:</b> This bit must be set low for normal operation.																																																																																																
5	RES	<b>RESERVED:</b> This bit must be set high for normal operation.																																																																																																
6	RES	<b>RESERVED:</b> This bit must be set low for normal operation.																																																																																																
7	SOFEN	<b>SOFTWARE ENABLE:</b> If this bit is set low then the user can program configuration registers A and B in software. If this bit is set high then the configuration registers are not accessible. If EECONFIG is high the configuration from the switches will be overwritten by the configuration from the EEPROM even if this bit is pulled high.																																																																																																

## 5.0 Register Descriptions (Continued)

### SIGNATURE REGISTER

This register is intended to allow the software programmer to determine which of the DP83907 family of devices and what board configuration is being used. This register is "hidden" behind a used register. Register 01H in the DP83907's Page 1 of registers is Physical Address Register 0, which is read/write. Now the Signature Register can be read at that address by following a read to 01H, on Page 1, with another read to 01H. If any other DP83907 register accesses take place between the two reads then the second read will access Physical Address Register 0. All writes are to Physical Address Register 0. The user can determine if the Signature Register exists by writing, while in Page 1, a known value to the Physical Address Register 0 then reading it back twice. The first read will be the known value. If the second read is not equal to the value written then the Signature Register exists. **This operation should only be attempted while the DP83907 is in STOP mode (STP bit in the Command Register set high).**

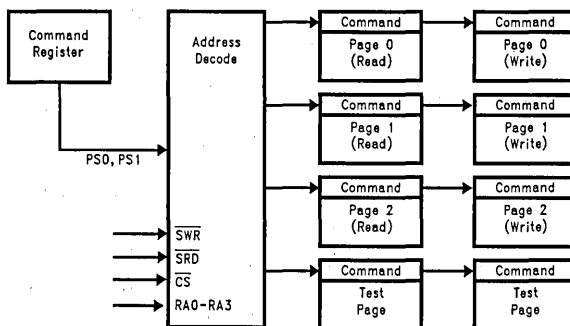


Bit	Symbol	Function								
0-3	REV0-3	<b>DP83907 FAMILY REVISION:</b> Shows that this is an DP83907 by holding the value: <table style="margin-left: 20px; border-collapse: collapse;"> <tr> <td style="padding-right: 10px;"><b>Rev 3</b></td> <td style="padding-right: 10px;"><b>Rev 2</b></td> <td style="padding-right: 10px;"><b>Rev 1</b></td> <td><b>Rev 0</b></td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> </tr> </table>	<b>Rev 3</b>	<b>Rev 2</b>	<b>Rev 1</b>	<b>Rev 0</b>	1	0	0	0
<b>Rev 3</b>	<b>Rev 2</b>	<b>Rev 1</b>	<b>Rev 0</b>							
1	0	0	0							
4	EEPR	<b>EEPROM PROGRAMMED:</b> If the upper byte of the EEPROM contains 073H when loaded into the DP83907 this bit will be high.								
0-3	SIG5-7 (MSA11-13)	<b>BOARD REVISION:</b> Shows that this board uses an NE2000 Architecture compatible interrupt structure: <table style="margin-left: 20px; border-collapse: collapse;"> <tr> <td style="padding-right: 10px;"><b>Sig 7</b></td> <td style="padding-right: 10px;"><b>Sig 6</b></td> <td><b>Sig 5</b></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> </tr> </table>	<b>Sig 7</b>	<b>Sig 6</b>	<b>Sig 5</b>	0	0	0		
<b>Sig 7</b>	<b>Sig 6</b>	<b>Sig 5</b>								
0	0	0								

## 5.0 Register Descriptions (Continued)

### 5.2 NIC CORE REGISTERS

All registers are 8-bit wide and mapped into two pages which are selected in the Command Register (PS0, PS1). Pins RA0-RA3 are used to address registers within each page. Page 0 registers are those registers which are commonly accessed during DP83907 operation while page 1 registers are used primarily for initialization. The registers are partitioned to avoid having to perform two write/read cycles to access commonly used registers.



TL/F/12082-18

FIGURE 23. NIC Core Register Mapping

### Register Assignments

#### Page 0 Address Assignments (PS1 = 0, PS0 = 0)

RA0-RA3	RD	WR
00H	Command (CR)	Command (CR)
01H	Current Local DMA Address 0 (CLDA0)	Page Start Register (PSTART)
02H	Current Local DMA Address 1 (CLDA1)	Page Stop Register (PSTOP)
03H	Boundary Pointer (BNRY)	Boundary Pointer (BNRY)
04H	Transmit Status Register (TSR)	Transmit Page Start Address (TPSR)
05H	Number of Collisions Register (NCR)	Transmit Byte Count Register 0 (TBCR0)
06H	FIFO (FIFO)	Transmit Byte Count Register 1 (TBCR1)
07H	Interrupt Status Register (ISR)	Interrupt Status Register (ISR)
08H	Current Remote DMA Address 0 (CRDA0)	Remote Start Address Register 0 (RSAR0)
09H	Current Remote DMA Address 1 (CRDA1)	Remote Start Address Register 1 (RSAR1)
0AH	Reserved	Remote Byte Count Register 0 (RBCR0)
0BH	Reserved	Remote Byte Count Register 1 (RBCR1)
0CH	Receive Status Register (RSR)	Receive Configuration Register (RCR)
0DH	Tally Counter 0 (Frame Alignment Errors) (CNTR0)	Transmit Configuration Register (TCR)
0EH	Tally Counter 1 (CRC Errors) (CNTR1)	Data Configuration Register (DCR)
0FH	Tally Counter 2 (Missed Packet Errors) (CNTR2)	Interrupt Mask Register (IMR)

## 5.0 Register Descriptions (Continued)

### Page 1 Address Assignments (PS1 = 0, PS0 = 1)

RA0-RA3	RD	WR
00H	Command (CR)	Command (CR)
01H	Physical Address Register 0 (PAR0)	Physical Address Register 0 (PA R0)
02H	Physical Address Register 1 (PAR1)	Physical Address Register 1 (PA R1)
03H	Physical Address Register 2 (PAR2)	Physical Address Register 2 (PA R2)
04H	Physical Address Register 3 (PAR3)	Physical Address Register 3 (PA R3)
05H	Physical Address Register 4 (PAR4)	Physical Address Register 4 (PA R4)
06H	Physical Address Register 5 (PAR5)	Physical Address Register 5 (PA R5)
07H	Current Page Register (CURR)	Current Page Register (C URR)
08H	Multicast Address Register 0 (MAR0)	Multicast Address Register 0 (MAR0)
09H	Multicast Address Register 1 (MAR1)	Multicast Address Register 1 (MAR1)
0AH	Multicast Address Register 2 (MAR2)	Multicast Address Register 2 (MAR2)
0BH	Multicast Address Register 3 (MAR3)	Multicast Address Register 3 (MAR3)
0CH	Multicast Address Register 4 (MAR4)	Multicast Address Register 4 (MAR4)
0DH	Multicast Address Register 5 (MAR5)	Multicast Address Register 5 (MAR5)
0EH	Multicast Address Register 6 (MAR6)	Multicast Address Register 6 (MAR6)
0FH	Multicast Address Register 7 (MAR7)	Multicast Address Register 7 (MAR7)

## 5.0 Register Descriptions (Continued)

### Page 2 Address Assignments (PS1 = 1, PS0 = 0)

RA0-RA3	RD	WR
00H	Command (CR)	Command (CR)
01H	Page Start Register (PSTART)	Current Local DMA Address 0 (CLDA0)
02H	Page Stop Register (PSTOP)	Current Local DMA Address 1 (CLDA1)
03H	Remote Next Packet Pointer	Remote Next Packet Pointer
04H	Transmit Page Start Address (TPSR)	Reserved
05H	Local Next Packet Pointer	Local Next Packet Pointer
06H	Address Counter (Upper)	Address Counter (Upper)
07H	Address Counter (Lower)	Address Counter (Lower)
08H	Reserved	Reserved
09H	Reserved	Reserved
0AH	Reserved	Reserved
0BH	Reserved	Reserved
0CH	Receive Configuration Register (RCR)	Reserved
0DH	Transmit Configuration Register (TCR)	Reserved
0EH	Data Configuration Register (DCR)	Reserved
0FH	Interrupt Mask Register (IMR)	Reserved

**Note:** Page 2 registers should only be accessed for diagnostic purposes. They should not be modified during normal operation.

Page 3 should never be modified.

## 5.0 Register Descriptions (Continued)

### COMMAND REGISTER (CR) 00H (READ/WRITE)

The Command Register is used to initiate transmissions, enable or disable Remote DMA operations and to select register pages. To issue a command the microprocessor sets the corresponding bit(s) (RD2, RD1, RD0, TXP). Further commands may be overlapped, but with the following rules: (1) If a transmit command overlaps with a remote DMA operation, bits RD0, RD1, and RD2 must be maintained for the remote DMA command when setting the TXP bit. Note, if a remote DMA command is re-issued when giving the transmit command, the DMA will complete immediately if the remote byte count register has not been re-initialized. (2) If a remote DMA operation overlaps a transmission, RD0, RD1, and RD2 may be written with the desired values and a "0" written to the TXP bit. Writing a "0" to this bit has no effect. (3) A remote write DMA may not overlap remote read operation or vice versa. Either of these operations must either complete or be aborted before the other operation may start. Bits PS1, PS0, RD2, and STP may be set any time.

7	6	5	4	3	2	1	0
PS1	PS0	RD2	RD1	RD0	TXP	STA	STP

Bit	Symbol	Description																								
D0	STP	<p><b>STOP:</b> Software reset command, takes the controller off-line, no packets will be received or transmitted. Any reception or transmission in progress will continue to completion before entering the reset state. To exit this state, the STP bit must be reset and the STA bit must be set high. To perform a software reset, this bit should be set high. The software reset has executed only when indicated by the RST bit in the ISR being set to 1. STP powers up high.</p> <p><b>Note:</b> If the DP83907 has previously been in start mode and the STP is set, both the STP and STA bits will remain set.</p>																								
D1	STA	<p><b>START:</b> This bit is used to activate the NIC Core after either power up, or when the NIC Core has been placed in a reset mode by software command or error. STA powers up low.</p>																								
D2	TXP	<p><b>TRANSMIT PACKET:</b> This bit must be set to initiate transmission of a packet. TXP is internally reset either after the transmission is completed or aborted. This bit should be set only after the Transmit Byte Count and Transmit Page Start registers have been programmed.</p>																								
D3-D5	RD0-RD2	<p><b>REMOTE DMA COMMAND:</b> These three encoded bits control operation of the Remote DMA channel. RD2 can be set to abort any Remote DMA command in progress. The Remote Byte Count Registers should be cleared when a Remote DMA has been aborted. The Remote Start Addresses are not restored to the starting address if the Remote DMA is aborted.</p> <table border="1"> <thead> <tr> <th>RD2</th> <th>RD1</th> <th>RD0</th> <th></th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>Not Allowed</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>Remote Read</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>Remote Write (Note 2)</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>Send Packet</td> </tr> <tr> <td>1</td> <td>X</td> <td>X</td> <td>Abort/Complete Remote DMA (Note 1)</td> </tr> </tbody> </table> <p><b>Note 1:</b> If a remote DMA operation is aborted and the remote byte count has not decremented to zero, the data transfer port should be read, for a remote read or send packet, or written to, for a remote write. This is required to ensure future correct operation.</p>	RD2	RD1	RD0		0	0	0	Not Allowed	0	0	1	Remote Read	0	1	0	Remote Write (Note 2)	0	1	1	Send Packet	1	X	X	Abort/Complete Remote DMA (Note 1)
RD2	RD1	RD0																								
0	0	0	Not Allowed																							
0	0	1	Remote Read																							
0	1	0	Remote Write (Note 2)																							
0	1	1	Send Packet																							
1	X	X	Abort/Complete Remote DMA (Note 1)																							
D6, D7	PS0, PS1	<p><b>PAGE SELECT:</b> These two encoded bits select which register page is to be accessed with addresses RA0-3.</p> <table border="1"> <thead> <tr> <th>PS1</th> <th>PS0</th> <th></th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Register Page 0</td> </tr> <tr> <td>0</td> <td>1</td> <td>Register Page 1</td> </tr> <tr> <td>1</td> <td>0</td> <td>Register Page 2</td> </tr> <tr> <td>1</td> <td>1</td> <td>Reserved</td> </tr> </tbody> </table>	PS1	PS0		0	0	Register Page 0	0	1	Register Page 1	1	0	Register Page 2	1	1	Reserved									
PS1	PS0																									
0	0	Register Page 0																								
0	1	Register Page 1																								
1	0	Register Page 2																								
1	1	Reserved																								

## 5.0 Register Descriptions (Continued)

### INTERRUPT STATUS REGISTER (ISR) 07H (READ/WRITE)

This register is accessed by the host processor to determine the cause of an interrupt. Any interrupt can be masked in the Interrupt Mask Register (IMR). Individual interrupt bits are cleared by writing a "1" into the corresponding bit of the ISR. The valid interrupt output is active as long as any unmasked signal is set, and will not go low until all unmasked bits in this register have been cleared. The ISR must be cleared after power up by writing it with all 1's.

7	6	5	4	3	2	1	0
RST	RDC	CNT	OVW	TXE	RXE	PTX	PRX

Bit	Symbol	Description
D0	PRX	<b>PACKET RECEIVED:</b> Indicates packet received with no errors.
D1	PTX	<b>PACKET TRANSMITTED:</b> Indicates packet transmitted with no errors.
D2	RXE	<b>RECEIVE ERROR:</b> Indicates that a packet was received with one or more of the following errors: —CRC Error —Frame Alignment Error —FIFO Overrun —Missed Packet
D3	TXE	<b>TRANSMIT ERROR:</b> Set when packet transmitted with one or more of the following errors: —Excessive Collisions —FIFO Underrun
D4	OVW	<b>OVERWRITE WARNING:</b> Set when receive buffer ring storage resources have been exhausted. (Local DMA has reached Boundary Pointer.)
D5	CNT	<b>COUNTER OVERFLOW:</b> Set when MSB of one or more of the Network Tally Counters has been set.
D6	RDC	<b>REMOTE DMA COMPLETE:</b> Set when Remote DMA operation has been completed.
D7	RST	<b>RESET STATUS:</b> Set when DP83907 enters reset state and cleared when a Start Command is issued to the CR. This bit is also set when a Receive Buffer Ring overflow occurs and is cleared when one or more packets have been removed from the ring. Writing to this bit has no effect. <b>Note:</b> This bit does not generate an interrupt, it is merely a status indicator.

## 5.0 Register Descriptions (Continued)

### INTERRUPT MASK REGISTER (IMR) 0FH (WRITE)

The Interrupt Mask Register is used to mask interrupts. Each interrupt mask bit corresponds to a bit in the Interrupt Status Register (ISR). If an interrupt mask bit is set, an interrupt will be issued whenever the corresponding bit in the ISR is set. If any bit in the IMR is set low, an interrupt will not occur when the bit in the ISR is set. **The IMR powers up all zeros.**

7	6	5	4	3	2	1	0
—	RDCE	CNTE	OVWE	TXEE	RXEE	PTXE	PRXE

Bit	Symbol	Description
D0	PRXE	<b>PACKET RECEIVED INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when packet received
D1	PTXE	<b>PACKET TRANSMITTED INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when packet is transmitted
D2	RXEE	<b>RECEIVE ERROR INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when packet received with error
D3	TXEE	<b>TRANSMIT ERROR INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when packet transmission results in error
D4	OVWE	<b>OVERWRITE WARNING INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when Buffer Management Logic lacks sufficient buffers to store incoming packet
D5	CNTE	<b>COUNTER OVERFLOW INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when MSB of one or more of the Network Statistics counters has been set
D6	RDCE	<b>DMA COMPLETE INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when Remote DMA transfer has been completed
D7	Reserved	Reserved



## 5.0 Register Descriptions (Continued)

### DATA CONFIGURATION REGISTER (DCR) 0EH (WRITE)

This Register is used to program the DP83907 for 8- or 16-bit memory interface, select byte ordering in 16-bit applications and establish FIFO thresholds. **The DCR must be initialized prior to loading the Remote Byte Count Registers. LAS is set on power up.**

7	6	5	4	3	2	1	0
—	FT1	FT0	ARM	LS	LAS	BOS	WTS

Bit	Symbol	Description																				
D0	WTS	<p><b>WORD TRANSFER SELECT</b></p> <p>0: Selects byte-wide DMA transfers            1: Selects word-wide DMA transfers            ; WTS establishes byte or word transfers for both Remote and Local DMA transfers</p> <p><b>Note:</b> When word-wide mode is selected, up to 32k words are addressable; A0 remains low.</p>																				
D1	BOS	<p><b>BYTE ORDER SELECT</b></p> <p>0: MS byte placed on AD15–AD8 and LS byte on AD7–AD0. (32xxx, 80x86)            1: MS byte placed on AD7–AD0 and LS byte on AD15–AD8. (680x0)            : Ignored when WTS is low</p>																				
D2	LAS	<p><b>LONG ADDRESS SELECT</b></p> <p>0: Dual 16-bit DMA mode            1: Single 32-bit DMA mode</p> <p>When LAS is high, the contents of the Remote DMA registers RSAR0, 1 are issued as A16–A31 Power up high</p>																				
D3	LS	<p><b>LOOPBACK SELECT</b></p> <p>0: Loopback mode selected. Bits D1 and D2 of the TCR must also be programmed for Loopback operation            1: Normal Operation</p>																				
D4	ARM	<p><b>AUTO-INITIALIZE REMOTE</b></p> <p>0: Send Command not executed, all packets removed from Buffer Ring under program control            1: Send Command executed, Remote DMA auto-initialized to remove packets from Buffer Ring</p> <p><b>Note:</b> Send Command cannot be used with 680x0 byte processors.</p>																				
D5 and D6	FT0 and FT1	<p><b>FIFO THRESHOLD SELECT:</b> Encoded FIFO threshold. Establishes point at which the memory bus is requested when filling or emptying the FIFO. During reception, the FIFO threshold indicates the number of bytes (or words) the FIFO has filled serially from the network before the FIFO is emptied onto the memory bus.</p> <p><b>Note:</b> FIFO threshold setting determines the DMA burst length.</p> <p><b>Receive Thresholds</b></p> <table border="1"> <thead> <tr> <th>FT1</th> <th>FT0</th> <th>Word Wide</th> <th>Byte Wide</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>1 Word</td> <td>2 Bytes</td> </tr> <tr> <td>0</td> <td>1</td> <td>2 Words</td> <td>4 Bytes</td> </tr> <tr> <td>1</td> <td>0</td> <td>4 Words</td> <td>8 Bytes</td> </tr> <tr> <td>1</td> <td>1</td> <td>6 Words</td> <td>12 Bytes</td> </tr> </tbody> </table> <p>During transmission, the FIFO threshold indicates the number of bytes (or words) the FIFO has filled from the Local DMA before being transferred to the memory. Thus, the transmission threshold is 16 bytes less the received threshold.</p>	FT1	FT0	Word Wide	Byte Wide	0	0	1 Word	2 Bytes	0	1	2 Words	4 Bytes	1	0	4 Words	8 Bytes	1	1	6 Words	12 Bytes
FT1	FT0	Word Wide	Byte Wide																			
0	0	1 Word	2 Bytes																			
0	1	2 Words	4 Bytes																			
1	0	4 Words	8 Bytes																			
1	1	6 Words	12 Bytes																			

## 5.0 Register Descriptions (Continued)

### TRANSMIT CONFIGURATION REGISTER (TCR) ODH (WRITE)

The transmit configuration establishes the actions of the transmitter section of the DP83907 during transmission of a packet on the network. **LB1 and LB0 which select loopback mode power up as 0.**

7	6	5	4	3	2	1	0
—	—	—	OFST	ATD	LB1	LB0	CRC

Bit	Symbol	Description																				
D0	CRC	<p><b>INHIBIT CRC</b></p> <p>0: CRC appended by transmitter 1: CRC inhibited by transmitter</p> <p>In loopback mode CRC can be enabled or disabled to test the CRC logic.</p>																				
D1 and D2	LB0 and LB1	<p><b>ENCODED LOOPBACK CONTROL:</b> These encoded configuration bits set the type of loopback that is to be performed. Note that loopback in mode 2 places the ENDEC Module in loopback mode and that D3 of the DCR must be set to zero for loopback operation.</p> <table border="1"> <thead> <tr> <th></th> <th>LB1</th> <th>LB0</th> <th></th> </tr> </thead> <tbody> <tr> <td>Mode 0</td> <td>0</td> <td>0</td> <td>Normal Operation (LPBK = 0)</td> </tr> <tr> <td>Mode 1</td> <td>0</td> <td>1</td> <td>Internal NIC Module Loopback (LPBK = 0)</td> </tr> <tr> <td>Mode 2</td> <td>1</td> <td>0</td> <td>Internal ENDEC Module Loopback (LPBK = 1)</td> </tr> <tr> <td>Mode 3</td> <td>1</td> <td>1</td> <td>External Loopback (LPBK = 0)</td> </tr> </tbody> </table>		LB1	LB0		Mode 0	0	0	Normal Operation (LPBK = 0)	Mode 1	0	1	Internal NIC Module Loopback (LPBK = 0)	Mode 2	1	0	Internal ENDEC Module Loopback (LPBK = 1)	Mode 3	1	1	External Loopback (LPBK = 0)
	LB1	LB0																				
Mode 0	0	0	Normal Operation (LPBK = 0)																			
Mode 1	0	1	Internal NIC Module Loopback (LPBK = 0)																			
Mode 2	1	0	Internal ENDEC Module Loopback (LPBK = 1)																			
Mode 3	1	1	External Loopback (LPBK = 0)																			
D3	ATD	<p><b>AUTO TRANSMIT DISABLE:</b> This bit allows another station to disable the DP83907'S transmitter by transmission of a particular multicast packet. The transmitter can be re-enabled by resetting this bit or by reception of a second particular multicast packet.</p> <p>0: Normal Operation 1: Reception of multicast address hashing to bit 62 disables transmitter, reception of multicast address hashing to bit 63 enables transmitter.</p>																				
D4	OFST	<p><b>COLLISION OFFSET ENABLE:</b> This bit modifies the backoff algorithm to allow prioritization of nodes.</p> <p>0: Backoff Logic implements normal algorithm. 1: Forces Backoff algorithm modification to 0 to <math>2^{\min(3 + n, 10)}</math> slot times for first three collisions, then follows standard backoff. (For the first three collisions, the station has higher average backoff delay making a low priority mode.)</p>																				
D5	Reserved	Reserved																				
D6	Reserved	Reserved																				
D7	Reserved	Reserved																				

## 5.0 Register Descriptions (Continued)

### TRANSMIT STATUS REGISTER (TSR) 04H (READ)

This register records events that occur on the media during transmission of a packet. It is cleared when the next transmission is initiated by the host. All bits remain low unless the event that corresponds to a particular bit occurs during transmission. Each transmission should be followed by a read of this register. The contents of this register are not specified until after the first transmission.

7	6	5	4	3	2	1	0
OWC	CDH	FU	CRS	ABT	COL	—	PTX

Bit	Symbol	Description
D0	PTX	<b>PACKET TRANSMITTED:</b> Indicates transmission without error. (No excessive collisions or FIFO underrun) (ABT = "0", FU = "0").
D1	Reserved	Reserved
D2	COL	<b>TRANSMIT COLLIDED:</b> Indicates that the transmission collided at least once with another station on the network. The number of collisions is recorded in the Number of Collisions Registers (NCR).
D3	ABT	<b>TRANSMIT ABORTED:</b> Indicates the DP83907 aborted transmission because of excessive collisions. (Total number of transmissions including original transmission attempt equals 16.)
D4	CRS	<b>CARRIER SENSE LOST:</b> This bit is set when carrier is lost during transmission of the packet. Transmission is not aborted on loss of carrier.
D5	FU	<b>FIFO UNDERRUN:</b> If the DP83907 cannot gain access of the bus before the FIFO empties, this bit is set. Transmission of the packet will be aborted.
D6	CDH	<b>CD HEARTBEAT:</b> Failure of the transceiver to transmit a collision signal after transmission of a packet will set this bit. The Collision Detect (CD) heartbeat signal must commence during the first 6.4 $\mu$ s of the Interframe Gap following a transmission. In certain collisions, the CD Heartbeat bit will be set even though the transceiver is not performing the CD heartbeat test.
D7	OWC	<b>OUT OF WINDOW COLLISION:</b> Indicates that a collision occurred after a slot time (51.2 $\mu$ s). Transmissions rescheduled as in normal collisions.

## 5.0 Register Descriptions (Continued)

### RECEIVE CONFIGURATION REGISTER (RCR) OCH (WRITE)

This register determines operation of the DP83907 during reception of a packet and is used to program what types of packets to accept.

7	6	5	4	3	2	1	0
—	—	MON	PRO	AM	AB	AR	SEP

Bits	Symbols	Description
D0	SEP	<b>SAVE ERRORED PACKETS</b> 0: Packets with receive errors are rejected. 1: Packets with receive errors are accepted. Receive errors are CRC and Frame Alignment errors.
D1	AR	<b>ACCEPT RUNT PACKETS:</b> This bit allows the receiver to accept packets that are smaller than 64 bytes. The packet must be at least 8 bytes long to be accepted as a runt. 0: Packets with fewer than 64 bytes rejected. 1: Packets with fewer than 64 bytes accepted.
D2	AB	<b>ACCEPT BROADCAST:</b> Enables the receiver to accept a packet with an all 1's destination address. 0: Packets with broadcast destination address rejected. 1: Packets with broadcast destination address accepted.
D3	AM	<b>ACCEPT MULTICAST:</b> Enables the receiver to accept a packet with a multicast address, all multicast addresses must pass the hashing array. 0: Packets with multicast destination address not checked. 1: Packets with multicast destination address checked.
D4	PRO	<b>PROMISCUOUS PHYSICAL:</b> Enables the receiver to accept all packets with a physical address. 0: Physical address of node must match the station address programmed in PAR0–PAR5. 1: All packets with physical addresses accepted.
D5	Reserved	Reserved (program to 0)
D6	Reserved	Reserved
D7	Reserved	Reserved

**Note:** D2 and D3 are "OR'd" together, i.e., if D2 and D3 are set the DP83907 will accept broadcast and multicast addresses as well as its own physical address. To establish full promiscuous mode, bits D2, D3, and D4 should be set. In addition the multicast hashing array must be set to all 1's in order to accept all multicast addresses.

## 5.0 Register Descriptions (Continued)

### RECEIVE STATUS REGISTER (RSR) OCH (READ)

This register records status of the received packet, including information on errors and the type of address match, either physical or multicast. The contents of this register are written to buffer memory by the DMA after reception of a good packet. If packets with errors are to be saved the receive status is written to memory at the head of the erroneous packet if an erroneous packet is received. If packets with errors are to be rejected the RSR will not be written to memory. The contents will be cleared when the next packet arrives. CRC errors, Frame Alignment errors and missed packets are counted internally by the DP83907 which relinquishes the Host from reading the RSR in real time to record errors for Network Management Functions. The contents of this register are not specified until after the first reception.

7	6	5	4	3	2	1	0
DFR	DIS	PHY	MPA	FO	FAE	CRC	PRX

Bit	Symbol	Description
D0	PRX	<b>PACKET RECEIVED INTACT:</b> Indicates packet received without error. (Bits CRC, FAE, FO, and MPA are zero for the received packet.)
D1	CRC	<b>CRC ERROR:</b> Indicates packet received with CRC error. Increments Tally Counter (CNTR1). This bit will also be set for Frame Alignment errors.
D2	FAE	<b>FRAME ALIGNMENT ERROR:</b> Indicates that the incoming packet did not end on a byte boundary and the CRC did not match at last byte boundary. Increments Tally Counter (CNTR0).
D3	FO	<b>FIFO OVERRUN:</b> This bit is set when the FIFO is not serviced causing overflow during reception. Reception of the packet will be aborted.
D4	MPA	<b>MISSED PACKET:</b> Set when packet intended for node cannot be accepted by SNIC because of a lack of receive buffers or if the controller is in monitor mode and did not buffer the packet to memory. Increments Tally Counter (CNTR2).
D5	PHY	<b>PHYSICAL/MULTICAST ADDRESS:</b> Indicates whether received packet had a physical or multicast address type. 0: Physical Address Match 1: Multicast/Broadcast Address Match
D6	DIS	<b>RECEIVER DISABLED:</b> Set when receiver disabled by entering Monitor mode. Reset when receiver is re-enabled when exiting Monitor mode.
D7	DFR	<b>DEFERRING:</b> Set when internal Carrier Sense or Collision signals are generated in the ENDEC module. If the transceiver has asserted the CD line as a result of the jabber, this bit will stay set indicating the jabber condition.

**Note:** Following coding applies to CRC and FAE bits:

FAE	CRC	Type of Error
0	0	No Error (Good CRC and <6 Dribble Bits)
0	1	CRC Error
1	0	Illegal, will not occur
1	1	Frame Alignment Error and CRC Error

## 5.0 Register Descriptions (Continued)

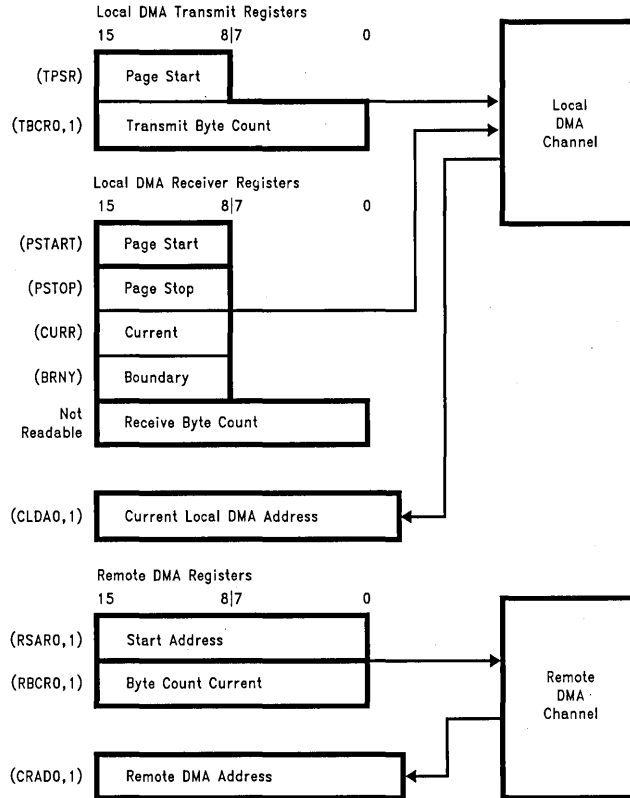


FIGURE 24. DMA Registers

TL/F/12082-19

### 5.3 DP8390 CORE DMA REGISTERS

The DMA Registers are partitioned into groups; Transmit, Receive and Remote DMA Registers. The Transmit registers are used to initialize the Local DMA Channel for transmission of packets while the Receive Registers are used to initialize the Local DMA Channel for packet Reception. The Page Stop, Page Start, Current and Boundary Registers are used by the Buffer Management Logic to supervise the Receive Buffer Ring. The Remote DMA Registers are used to initialize the Remote DMA

**Note:** In the figure above, registers are shown as 8 or 16 bits wide. Although some registers are 16-bit internal registers, all registers are accessed as 8-bit registers. Thus the 16-bit Transmit Byte Count Register is broken into two 8-bit registers. TBCR0 TBCR1. Also TPSR, PSTART, PSTOP, CURR and BRNY only check or control the upper 8 bits of address information on the bus. Thus they are shifted to positions 15-8 in the diagram above.

#### Transmit DMA Registers

##### TRANSMIT PAGE START REGISTER (TPSR)

This register points to the assembled packet to be transmitted. Only the eight higher order addresses are specified since all transmit packets are assembled on 256-byte page

boundaries. The bit assignment is shown below. The values placed in bits D7-D0 will be used to initialize the higher order address (A8-A15) of the Local DMA for transmission. The lower order bits (A7-A0) are initialized to zero.

#### Bit Assignment

	7	6	5	4	3	2	1	0
TPSR	A15	A14	A13	A12	A11	A10	A9	A8

(A7-A0 Initialized to 0)

##### TRANSMIT BYTE COUNT REGISTER 0,1 (TBCR0, TBCR1)

These two registers indicate the length of the packet to be transmitted in bytes. The count must include the number of bytes in the source, destination, length and data fields. The maximum number of transmit bytes allowed is 64K bytes. The DP83907 will not truncate transmissions longer than 1500 bytes. The bit assignment is shown below:

	7	6	5	4	3	2	1	0
TBCR1	L15	L14	L13	L12	L11	L10	L9	L8
	7	6	5	4	3	2	1	0
TBCR0	L7	L6	L5	L4	L3	L2	L1	L0

## 5.0 Register Descriptions (Continued)

### Local DMA Receive Registers

#### PAGE START STOP REGISTERS (PSTART, PSTOP)

The Page Start and Page Stop Registers program the starting and stopping address of the Receive Buffer Ring. Since the DP83907 uses fixed 256-byte buffers aligned on page boundaries only the upper eight bits of the start and stop address are specified.

#### PSTART, PSTOP bit assignment

	7	6	5	4	3	2	1	0
PSTART	A15	A14	A13	A12	A11	A10	A9	A8
PSTOP								

#### BOUNDARY (BNRY) REGISTER

This register is used to prevent overflow of the Receive Buffer Ring. Buffer management compares the contents of this register to the next buffer address when linking buffers together. If the contents of this register match the next buffer address the Local DMA operation is aborted.

	7	6	5	4	3	2	1	0
BNRY	A15	A14	A13	A12	A11	A10	A9	A8

#### CURRENT PAGE REGISTER (CURR)

This register is used internally by the Buffer Management Logic as a backup register for reception. CURR contains the address of the first buffer to be used for a packet reception and is used to restore DMA pointers in the event of receive errors. This register is initialized to the same value as PSTART and should not be written to again unless the controller is Reset.

	7	6	5	4	3	2	1	0
CURR	A15	A14	A13	A12	A11	A10	A9	A8

#### CURRENT LOCAL DMA REGISTER 0,1 (CLDA0,1)

These two registers can be accessed to determine the current Local DMA Address.

	7	6	5	4	3	2	1	0
CLDA1	A15	A14	A13	A12	A11	A10	A9	A8

	7	6	5	4	3	2	1	0
CLDA0	A7	A6	A5	A4	A3	A2	A1	A0

### Remote DMA Registers

#### REMOTE START ADDRESS REGISTERS (RSAR0,1)

Remote DMA operations are programmed via the Remote Start Address (RSAR0,1) and Remote Byte Count (RBCR0,1) registers. The Remote Start Address is used to point to the start of the block of data to be transferred and the Remote Byte Count is used to indicate the length of the block (in bytes).

	7	6	5	4	3	2	1	0
RSAR1	A15	A14	A13	A12	A11	A10	A9	A8

	7	6	5	4	3	2	1	0
RSAR0	A7	A6	A5	A4	A3	A2	A1	A0

#### REMOTE BYTE COUNT REGISTERS (RBC0,1)

	7	6	5	4	3	2	1	0
RBCR1	A15	A14	A13	A12	A11	A10	A9	A8

	7	6	5	4	3	2	1	0
RBCR0	A7	A6	A5	A4	A3	A2	A1	A0

#### Note:

RSAR0 programs the start address bits A0–A7.

RSAR1 programs the start address bits A8–A15.

Address incremented by two for word transfers, and by one for byte transfers. Byte count decremented by two for word transfers and by one for byte transfers.

RBCR0 programs LSB byte count.

RBCR1 programs MSB byte count.

#### CURRENT REMOTE DMA ADDRESS (CRDA0, CRDA1)

The Current Remote DMA Registers contain the current address of the Remote DMA. The bit assignment is shown below:

	7	6	5	4	3	2	1	0
CRDA1	A15	A14	A13	A12	A11	A10	A9	A8

	7	6	5	4	3	2	1	0
CRDA0	A7	A6	A5	A4	A3	A2	A1	A0

#### Physical Address Registers (PAR0–PAR5)

The physical address registers are used to compare the destination address of incoming packets for rejecting or accepting packets. Comparisons are performed on a byte-wide basis. The bit assignment shown below relates the sequence in PAR0–PAR5 to the bit sequence of the received packet.

	D7	D6	D5	D4	D3	D2	D1	D0
PAR0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
PAR1	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
PAR2	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
PAR3	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
PAR4	DA39	DA38	DA37	DA36	DA35	DA34	DA33	DA32
PAR5	DA47	DA46	DA45	DA44	DA43	DA42	DA41	DA40

	Destination Address						Source			
	P/S	DA0	DA1	DA2	DA3	...	DA46	DA47	SA0	...

#### Note:

P/S = Preamble, Synchron

DA0 = Physical/Multicast Bit

#### Multicast Address Registers (MAR0–MAR7)

The multicast address registers provide filtering of multicast addresses hashed by the CRC logic. All destination addresses are fed through the CRC logic and as the last bit of the destination address enters the CRC, the 6 most significant bits of the CRC generator are latched. These 6 bits are then decoded by a 1 of 64 decode to index a unique filter bit (FB0–63) in the multicast address registers. If the filter bit selected is set, the multicast packet is accepted. The sys-

## 5.0 Register Descriptions (Continued)

tem designer would use a program to determine which filter bits to set in the multicast registers. All multicast filter bits that correspond to multicast address accepted by the node are then set to one. To accept all multicast packets all of the registers are set to all ones.

**Note:** Although the hashing algorithm does not guarantee perfect filtering of multicast address, it will perfectly filter up to 64 multicast addresses if these addresses are chosen to map into unique locations in the multicast filter.

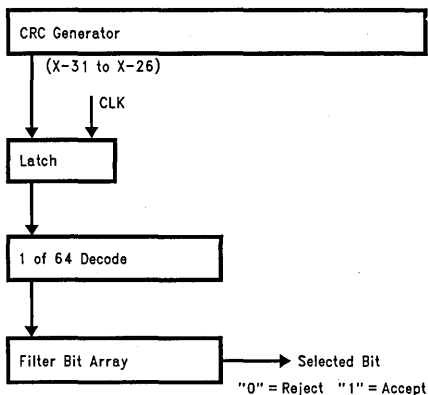


FIGURE 25. Multicast Addressing

TL/F/12082-20

## 6.0 Operation of DP83907

This section details the operation of the DP83907. The operations discussed are packet reception and transmission, bus operations, and loopback diagnostics.

### 6.1 TRANSMIT/RECEIVE PACKET ENCAPSULATION/DECAPSULATION;

A standard IEEE 802.3 packet consists of the following fields: preamble, Start of Frame Delimiter (SFD), destination address, source address, length, data, and Frame Check Sequence (FCS). The typical format is shown in the figure following. The packets are Manchester encoded and decoded by the ENDEC module and transferred serially to the NIC module using NRZ data with a clock. All fields are of fixed length except for the data field. The DP83907 generates and appends the preamble, SFD and FCS field during transmission. The Preamble and SFD fields are stripped during reception. (The CRC is passed through to buffer memory during reception.)

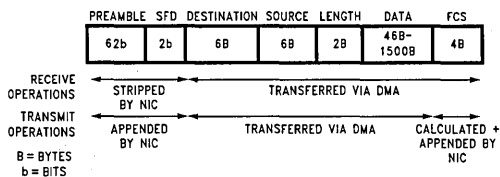


FIGURE 26. Ethernet Packet

TL/F/12082-21

### Preamble and Start of Frame Delimiter (SFD)

The Manchester encoded alternating 1, 0 preamble field is used by the ENDEC to acquire bit synchronization with an incoming packet. When transmitted each packet contains 62 bits of alternating 1, 0 preamble. Some of this preamble will be lost as the packet travels through the network. The preamble field is stripped by the NIC module. Byte alignment is performed with the Start of Frame Delimiter (SFD) pattern which consists of two consecutive 1's. The DP83907 does not treat the SFD pattern as a byte, it detects only the two bit pattern. This allows any preceding preamble within the SFD to be used for phase locking.

### Destination Address

The destination address indicates the destination of the packet on the network and is used to filter unwanted packets from reaching a node. There are three types of address formats supported by the DP83907: physical, multicast and broadcast. The physical address is a unique address that corresponds only to a single node. All physical addresses have an MSB of "0". These addresses are compared to the internally stored physical address registers. Each bit in the destination address must match in order for the DP83907 to accept the packet. Multicast addresses begin with an MSB of "1". The DP83907 filters multicast addresses using a standard hashing algorithm that maps all multicast addresses into a 6-bit value. This 6-bit value indexes a 64-bit array that filters the value. If the address consists of all 1's it is a broadcast address, indicating that the packet is intended for all nodes. A promiscuous mode allows reception of all packets: the destination address is not required to match any filters. Physical, broadcast, multicast, and promiscuous address modes can be selected.

### Source Address

The source address is the physical address of the node that sent the packet. Source addresses cannot be multicast or broadcast addresses. This field is simply passed to buffer memory.

### Length Field

The 2-byte length field indicates the number of bytes that are contained in the data field of the packet. This field is not interpreted by the DP83907.

### Data Field

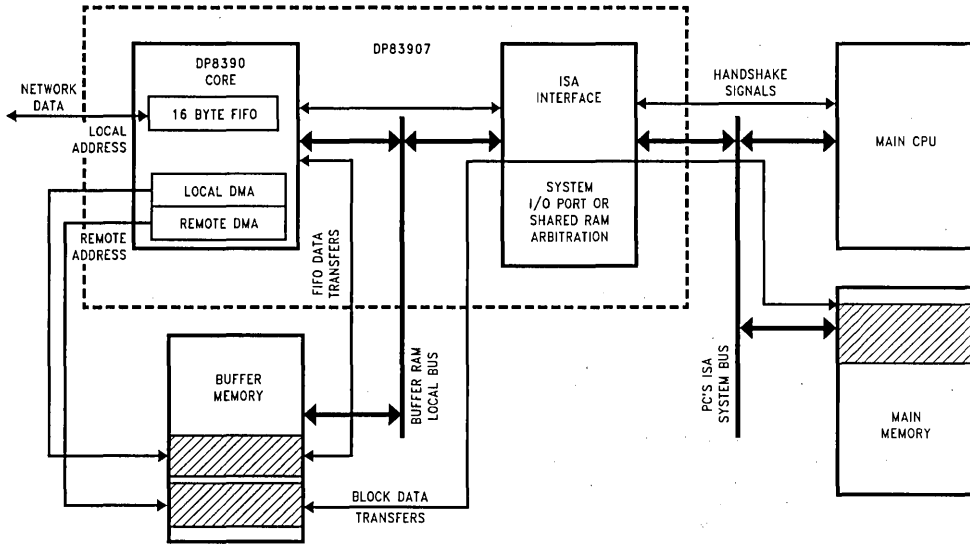
The data field consists of anywhere from 46 to 1500 bytes. Messages longer than 1500 bytes need to be broken into multiple packets. Messages shorter than 46 bytes will require appending a pad to bring the data field to the minimum length of 46 bytes. If the data field is padded, the number of valid data bytes is indicated in the length field. **The DP83907 does not strip or append pad bytes for short packets, or check for oversize packets.**

### FCS Field

The Frame Check Sequence (FCS) is a 32-bit CRC field calculated and appended to a packet during transmission to allow detection of errors when a packet is received. During reception, error free packets result in a specific pattern in the CRC generator. Packets with improper CRC will be rejected. The AUTODIN II ( $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X^1 + 1$ ) polynomial is used for the CRC calculations.



### 6.0 Operation of DP83907 (Continued)



TL/F/12082-22

FIGURE 27. DP83907 Bus Architecture

#### 6.2 BUFFER MEMORY ACCESS CONTROL (DMA)

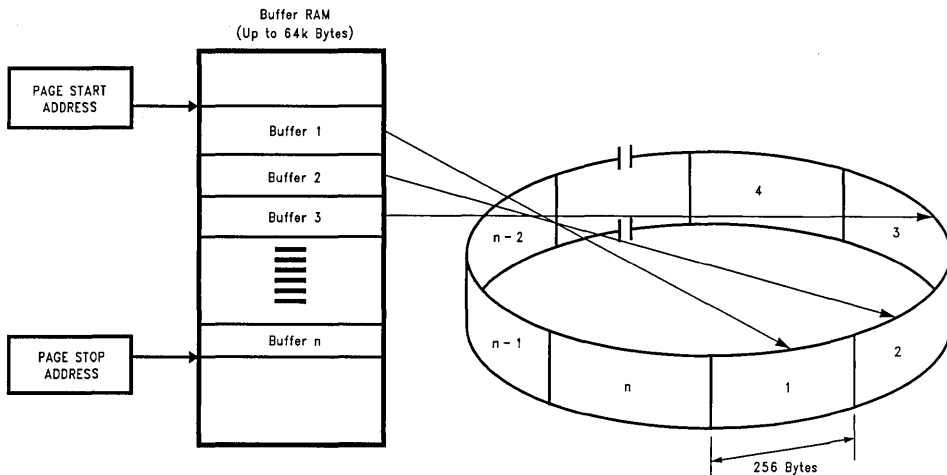
The buffer memory control capabilities of the DP83907 greatly simplify the use of the DP83907 in typical configurations. The local DMA channel transfers data between the FIFO and memory. On transmission, the packet is DMA'd from memory to the FIFO in bursts. Should a collision occur (up to 15 times), the packet is re-transmitted with no processor intervention. On reception, packets are moved via DMA from the FIFO to the receive buffer ring (as explained below).

A Remote DMA channel is also provided on the DP83907 to accomplish transfers between a buffer memory and an internal Data Port when using the DP83907 in I/O Mode. This

Remote DMA channel is not used when the DP83907 is used in a shared Memory mode. In this second mode the buffer memory is dual ported, and directly mapped into the system memory. In this mode the system CPU directly accesses the RAM under software control to transfer packet data.

The following sections describe the operation of the Local DMA channel for packet reception which is used in both modes. For Shared Memory mode the description of the Remote DMA does not apply.

For reference an example configuration using the DP83907 is shown in Figure 27.



TL/F/12082-23

FIGURE 28. DP83907 Receive Buffer Ring

## 6.0 Operation of DP83907 (Continued)

### 6.3 PACKET RECEPTION

The Local DMA receive channel uses a Buffer Ring Structure comprised of a series of contiguous fixed length 256 byte (128 word) buffers for storage of received packets. The location of the Receive Buffer Ring is programmed in two registers, a Page Start and a Page Stop Register. Ethernet packets consist of a distribution of shorter link control packets and longer data packets, the 256 byte buffer length provides a good compromise between short packets and longer packets to most efficiently use memory. In addition these buffers provide memory resources for storage of back-to-back packets in loaded networks. The assignment of buffers for storing packets is controlled by Buffer Management Logic in the DP83907. The Buffer Management Logic provides three basic functions: linking receive buffers for long packets, recovery of buffers when a packet is rejected, and recirculation of buffer pages that have been read by the host.

At initialization, a portion of the 64k byte (or 32k word) address space is reserved for the receive buffer ring. Two eight bit registers, the Page Start Address Register (PSTART) and the Page Stop Address Register (PSTOP) define the physical boundaries of where the buffers reside. The DP83907 treats the list of buffers as a logical ring; whenever the DMA address reaches the Page Stop Address, the DMA is reset to the Page Start Address.

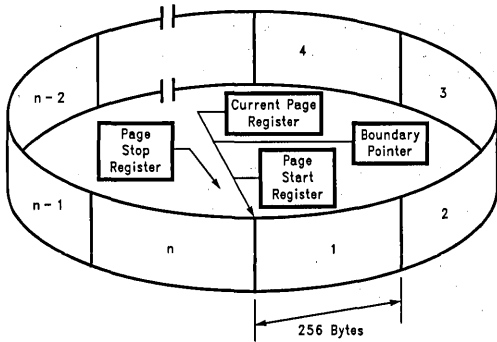


FIGURE 29. Buffer Ring at Initialization

### Initialization of the Buffer Ring

Two static registers and two working registers control the operation of the Buffer Ring. These are the Page Start Register, Page Stop Register (both described previously), the Current Page Register and the Boundary Pointer Register. The Current Page Register points to the first buffer used to store a packet and is used to restore the DMA for writing status to the Buffer Ring or for restoring the DMA address in the event of a Runt packet, a CRC, or Frame Alignment error. The Boundary Register points to the first packet in the Ring not yet read by the host. If the local DMA address ever reaches the Boundary, reception is aborted. The Boundary Pointer is also used to initialize the Remote DMA for removing a packet and is advanced when a packet is removed. A simple analogy to remember the function of these registers is that the Current Page Register acts as a Write Pointer and the Boundary Pointer acts as a Read Pointer.

**Note 1:** At initialization, the Page Start Register value should be loaded into both the Current Page Register and the Boundary Pointer Register, if using the Send Packet Command. If using manual Remote Reads, the Current Page Register must always remain 1 + Boundary Register.

**Note 2:** The Page Start Register must not be initialized to 00H.

### Beginning of Reception

When the first packet begins arriving the DP83907 begins storing the packet at the location pointed to by the Current Page Register. An offset of 4 bytes is saved in this first buffer to allow room for storing receive status corresponding to this packet.

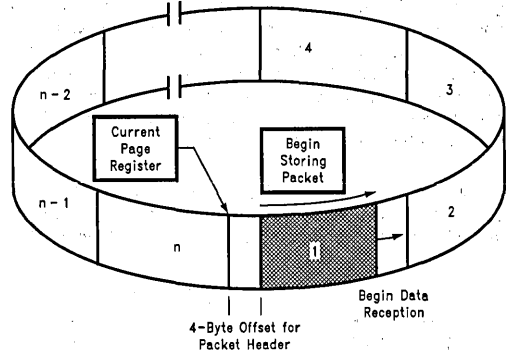


FIGURE 30. Received Packet Enters the Buffer Pages

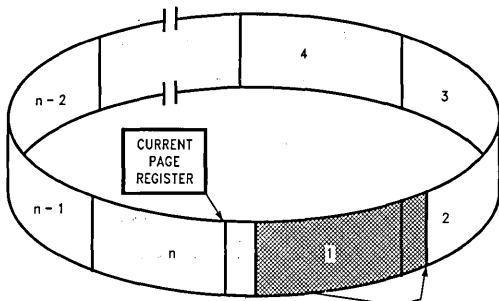
## 6.0 Operation of DP83907 (Continued)

### Linking Receive Buffer Pages

If the length of the packet exhausts the first 256 byte buffer, the DMA performs a forward link to the next buffer to store the remainder of the packet. For a maximal length packet the buffer logic will link six buffers to store the entire packet. Buffers cannot be skipped when linking, a packet will always be stored in contiguous buffers. Before the next buffer can be linked, the Buffer Management Logic performs two comparisons. The first comparison tests for equality between the DMA address of the next buffer and the contents of the Page Stop Register. If the buffer address equals the Page Stop Register, the buffer management logic will restore the DMA to the first buffer in the Receive Buffer Ring value programmed in the Page Start Address Register. The second comparison tests for equality between the DMA address of the next buffer address and the contents of the Boundary Pointer Register. If the two values are equal the reception is aborted. The Boundary Pointer Register can be used to protect against overwriting any area in the receive buffer ring that has not yet been read. When linking buffers, buffer management will never cross this pointer, effectively avoiding any overwrites. If the buffer address does not match either the Boundary Pointer or Page Stop Address, the link to the next buffer is performed.

### Linking Buffers

Before the DMA can enter the next contiguous 256 byte buffer, the address is checked for equality to PSTOP and to the Boundary Pointer. If neither are reached, the DMA is allowed to use the next buffer.



- 1) Check for = to PSTOP
- 2) Check for = to Boundary

TL/F/12082-26

FIGURE 31. Linking Receive Buffer Pages

### Buffer Ring Overflow

If the Buffer Ring has been filled and the DMA reaches the Boundary Pointer Address, reception of the incoming packet will be aborted by the DP83907. Thus, the packets previously received and still contained in the Ring will not be destroyed.

In heavily loaded networks which cause overflows of the Receive Buffer Ring, the DP83907 may disable the local DMA and suspend further receptions even if the Boundary register is advanced beyond the Current register. In the event that the DP83907 should encounter a receive buffer overflow, it is necessary to implement the following routine. A receive buffer overflow is indicated by the DP83907's assertion of the overflow bit (OVW) in the Interrupt Status Register (ISR).

If this routine is not adhered to, the DP83907 may act in an unpredictable manner. It should also be noted that it is not permissible to service an overflow interrupt by continuing to empty packets from the receive buffer without implementing the prescribed overflow routine. A flow chart of the DP83907's overflow routine can be found in *Figure 32*.

**Note:** It is necessary to define a variable in the driver, which will be called "Resend".

- 1) Read and store the value of the TXP bit in the DP83907's Command Register.
- 2) Issue the STOP command to the DP83907. This is accomplished by setting the STP bit in the DP83907's Command Register. Writing 21H to the Command Register will stop the DP83907.
- 3) Wait for at least 1.6 ms. Since the DP83907 will complete any transmission or reception that is in progress, it is necessary to time out for the maximum possible duration of an Ethernet transmission or reception. By waiting 1.6 ms this is achieved with some guard band added. Previously, it was recommended that the RST bit of the Interrupt Status Register be polled to insure that the pending transmission or reception is completed. This bit is not a reliable indicator and subsequently should be ignored.

## 6.0 Operation of DP83907 (Continued)

4) Clear the DP83907's Remote Byte Count registers (RBCR0 and RBCR1).

5) Read the stored value of the TXP bit from step 1, above.

If this value is a 0, set the "Resend" variable to a 0 and jump to step 6.

If this value is a 1, read the DP83907's Interrupt Status Register. If either the Packet Transmitted bit (PTX) or Transmit Error bit (TXE) is set to a 1, set the "Resend" variable to a 0 and jump to step 6. If neither of these bits is set, place a 1 in the "Resend" variable and jump to step 6.

This step determines if there was a transmission in progress when the stop command was issued in step 2. If there was a transmission in progress, the DP83907's ISR is read to determine whether or not the packet was recognized by the DP83907. If neither the PTX nor TXE bit was set, then the packet will essentially be lost and retransmitted only after a time-out takes place in the upper level software. By determining that the packet was lost at the driver level, a transmit command can be reissued to the DP83907 once the overflow routine is completed (as in step 11). Also, it is possible for the DP83907 to defer indefinitely, when it is stopped on a busy network. Step 5 also alleviates this problem. Step 5 is essential and should not be omitted from the overflow routine, in order for the DP83907 to operate correctly.

6) Place the DP83907 in either mode 1 or mode 2 loopback. This can be accomplished by setting bits D2 and D1, of the Transmit Configuration Register, to "0,1" or "1,0", respectively.

7) Issue the START command to the DP83907. This can be accomplished by writing 22H to the Command Register. This is necessary to activate the DP83907's Remote DMA channel.

8) Remove one or more packets from the receive buffer ring.

9) Reset the overwrite warning (OVW, overflow) bit in the Interrupt Status Register.

10) Take the DP83907 out of loopback. This is done by writing the Transmit Configuration Register with the value it contains during normal operation. (Bits D2 and D1 should both be programmed to 0.)

11) If the "Resend" variable is set to a 1, reset the "Resend" variable and reissue the transmit command. This is done by writing a value of 26H to the Command Register. If the "Resend" variable is 0, nothing needs to be done.

**Note 1:** If Remote DMA is not being used, the DP83907 does not need to be started before packets can be removed from the receive buffer ring. Hence, step 8 could be done before step 7, eliminating or reducing the time spent polling in step 5.

**Note 2:** When the DP83907 is in STOP mode, the Missed Packet Tally counter is disabled.

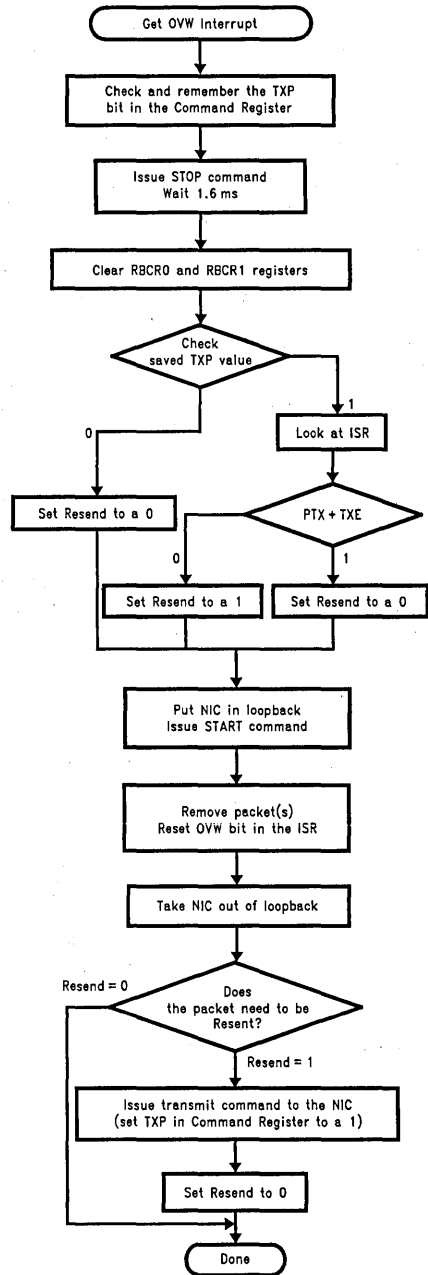
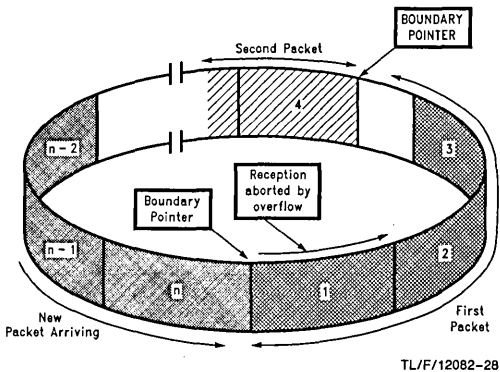


FIGURE 32. Overflow Routine

TL/F/12082-27

## 6.0 Operation of DP83907 (Continued)



**FIGURE 33. Received Packet Aborted If It Hits Boundary**

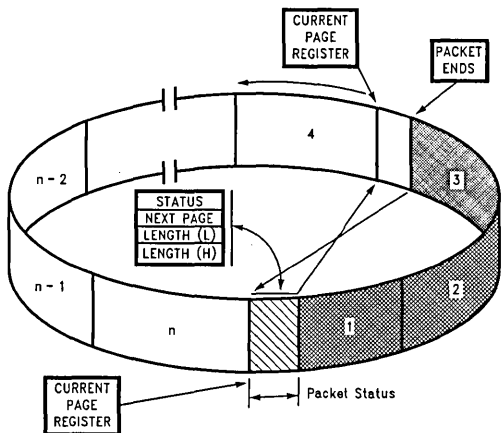
### Enabling the DP83907 on an Active Network

After the DP83907 has been initialized the procedure for disabling and then re-enabling the DP83907 on the network is similar to handling Receive Buffer Ring overflow as described previously.

- 1) Program Command Register for page 0 (Command Register = 21H)
- 2) Initialize Data Configuration Register (DCR)
- 3) Clear Remote Byte Count Registers (RBCR0, RBCR1) if using Remote DMA.
- 4) Initialize Receive Configuration Register (RCR)
- 5) Place the DP83907 in LOOPBACK mode 1 or 2 (Transmit Configuration Register = 02H or 04H)
- 6) Initialize Receive Buffer Ring: Boundary Pointer (BNDRY), Page Start (PSTART), and Page Stop (PSTOP)
- 7) Clear Interrupt Status Register (ISR) by writing OFFH to it.
- 8) Initialize Interrupt Mask Register (IMR)
- 9) Program Command Register for page 1 (Command Register = 61H)
  - i) Initialize Physical Address Registers (PAR0-PAR5)
  - ii) Initialize Multicast Address Registers (MAR0-MAR7)
  - iii) Initialize CURRENT pointer
- 10) Put DP83907 in START mode (Command Register = 22H). The local receive DMA is still not active since the DP83907 is in LOOPBACK.
- 11) Initialize the Transmit Configuration for the intended value. The DP83907 is now ready for transmission and reception.

### End of Packet Operations

At the end of the packet the DP83907 determines whether the received packet is to be accepted or rejected. It either branches to a routine to store the Buffer Header or to another routine that recovers the buffers used to store the packet.



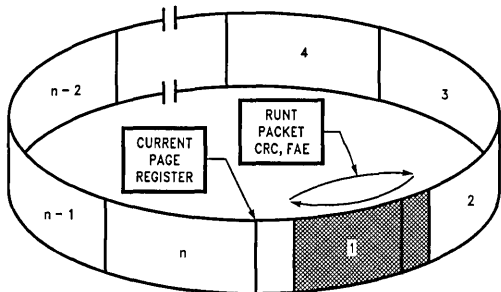
**FIGURE 34. Termination of Received Packet-Packet Accepted**

### Successful Reception

If the packet is successfully received, the DMA is restored to the first buffer used to store the packet (pointed to by the Current Page Register). The DMA then stores the Receive Status, a Pointer to where the next packet will be stored (Buffer 4) and the number of received bytes. Note that the remaining bytes in the last buffer are discarded and reception of the next packet begins on the next empty 256-byte buffer boundary. The Current Page Register is then initialized to the next available buffer in the Buffer Ring. (The location of the next buffer had been previously calculated and temporarily stored in an internal scratchpad register.)

### Buffer Recovery for Rejected Packets

If the packet is a runt packet or contains CRC or Frame Alignment errors, it is rejected. The buffer management logic resets the DMA back to the first buffer page used to store the packet (pointed to by CURR), recovering all buffers that had been used to store the rejected packet. This operation will not be performed if the DP83907 is programmed to accept either runt packets or packets with CRC or Frame Alignment errors. The received CRC is always stored in buffer memory after the last byte of received data for the packet.



**FIGURE 35. Termination of Receive Packet-Packet Reject**

## 6.0 Operation of DP83907 (Continued)

### Error Recovery

If the packet is rejected as shown, the DMA is restored by the DP83907 by reprogramming the DMA starting address pointed to by the Current Page Register.

### Storage Format for Received Packets

The following diagrams describe the format for how received packets are placed into memory by the local DMA channel. These modes are selected in the Data Configuration Register.

AD15	AD8	AD7	AD0
Next Packet Pointer		Receive Status	
Receive Byte Count 1		Receive Byte Count 0	
Byte 2		Byte 1	

BOS = 0, WTS = 1 in Data Configuration Register. This format is used with Series 32xxx, or 808xx processors.

AD15	AD8	AD7	AD0
Next Packet Pointer		Receive Status	
Receive Byte Count 0		Receive Byte Count 1	
Byte 1		Byte 2	

BOS = 1, WTS = 1 in Data Configuration Register. This format is used with 680 x 0 type processors. (Note: the Receive Count ordering remains the same for BOS = 0 or 1.)

Receive Status
Next Packet Pointer
Receive Byte Count 0
Receive Byte Count 1
Byte 0
Byte 1

BOS = 0 WTS = 0 in Data Configuration Register. This format is used with general 8-bit processors.

### 6.4 PACKET TRANSMISSION

The Local DMA is also used during transmission of a packet. Three registers control the DMA transfer during transmission, a Transmit Page Start Address Register (TPSR) and the Transmit Byte Count Registers (TBCR0, 1). When the DP83907 receives a command to transmit the packet pointed to by these registers, buffer memory data will be moved into the FIFO as required during transmission. The DP83907 will generate and append the preamble, synch and CRC fields.

### General Transmit Packet Format

Transmit	Destination Address	6 Bytes
Byte	Source Address	6 Bytes
Count	Type/Length	2 Bytes
TBCR0, 1	Data Pad (if data < 46 Bytes)	≥ 46 Bytes

### Transmit Packet Assembly

The DP83907 requires a contiguous assembled packet with the format shown. The transmit byte count includes the Destination Address, Source Address, Length Field and Data. It does not include preamble and CRC. When transmitting data smaller than 46 bytes, the packet must be padded to a minimum size of 64 bytes. The programmer is responsible for adding and stripping pad bytes.

The packets are placed in the buffer RAM by the system. In I/O Mode the system programs the NIC Core's Remote DMA to mode the data from the data port to the RAM handshaking with system transfers loading the I/O data port. In Shared Memory Mode the packets are written directly to the RAM by system using standard memory transfer instructions (MOV).

For I/O mode the data transfer must be 16 bits (1 word) when in 16 bit mode, and 8 bits when the DP83907 is set in 8 bit mode. The data width is selected by setting the WTS bit in the Data Configuration Register and setting the DWID pin for the proper mode.

In Shared Memory mode data transfer can be accomplished by using either 8- or 16-bit data transfer instructions, because this mode responds to 8/16 bit data signaling on the ISA bus. In this mode Shared Memory Control Register 2 bit 6 sets the bus interface data width, and the NIC Core's data width is set by the WTS bit in the Data Configuration Register.

### Transmission

Prior to transmission, the TPSR (Transmit Page Start Register) and TBCR0, TBCR1 (Transmit Byte Count Registers) must be initialized. To initiate transmission of the packet the TXP bit in the Command Register is set. The Transmit Status Register (TSR) is cleared and the DP83907 begins to prefetch transmit data from memory (unless the DP83907 is currently receiving). If the interframe gap has timed out the DP83907 will begin transmission.

## 6.0 Operation of DP83907 (Continued)

### Conditions Required to Begin Transmission

In order to transmit a packet, the following three conditions must be met:

1. The Interframe Gap Timer has timed out the first 6.4  $\mu$ s of the Interframe Gap
2. At least one byte has entered the FIFO. (This indicates that the burst transfer has been started)
3. If a collision had been detected then before transmission the packet time must have timed out.

In typical systems the DP83907 prefetches the first burst of bytes before the 6.4  $\mu$ s timer expires. The time during which DP83907 transmits preamble can also be used to load the FIFO.

**Note:** If carrier sense is asserted before a byte has been loaded into the FIFO, the DP83907 will become a receiver.

### Collision Recovery

During transmission, the Buffer Management logic monitors the transmit circuitry to determine if a collision has occurred. If a collision is detected, the Buffer Management logic will reset the FIFO and restore the Transmit DMA pointers for retransmission of the packet. The COL bit will be set in the TSR and the NCR (Number of Collisions Register) will be incremented. If 15 retransmissions each result in a collision the transmission will be aborted and the ABT bit in the TSR will be set.

**Note:** NCR reads as zeroes if excessive collisions are encountered.

### Transmit Packet Assembly Format

The following diagrams describe the format for how packets must be assembled prior to transmission for different byte ordering schemes. The various formats are selected in the Data Configuration Register.

D15	D8	D7	D0
Destination Address 1	Destination Address 0		
Destination Address 3	Destination Address 2		
Destination Address 5	Destination Address 4		
Source Address 1	Source Address 0		
Source Address 3	Source Address 2		
Source Address 5	Source Address 4		
Type/Length 1	Type Length 0		
Data 1	Data 0		

BOS = 0, WTS = 1 in Data Configuration Register.

This format is used with Series 32xxx, or 808xx processors.

D15	D8	D7	D0
Destination Address 0		Destination Address 1	
Destination Address 2		Destination Address 3	
Destination Address 4		Destination Address 5	
Source Address 0		Source Address 1	
Source Address 2		Source Address 3	
Source Address 4		Source Address 5	
Type Length 0		Type/Length 1	
Data 0		Data 1	

BOS = 1, WTS = 1 in Data Configuration Register.

This format is used with 680x0 type processors.

D7	D0
Destination Address 0	
Destination Address 1	
Destination Address 2	
Destination Address 3	
Destination Address 4	
Destination Address 5	
Source Address 0	
Source Address 1	
Source Address 2	
Source Address 3	
Source Address 4	
Source Address 5	

BOS = 0, WTS = 0 in Data Configuration Register.

This format is used with general 8 bit processors.

**Note:** All examples above will result in a transmission of a packet in order of DA0, DA1, DA3... bits within each byte will be transmitted least significant bit first.

DA = Destination Address

## 6.0 Operation of DP83907 (Continued)

### 6.5 LOOPBACK DIAGNOSTICS

Three forms of local loopback are provided on the DP83907. The user has the ability to loopback through the deserializer on the controller, through the ENDEC module or transceiver. **Because of the half duplex architecture of the DP83907, loopback testing is a special mode of operation with the following restrictions:**

#### Restrictions During Loopback

The FIFO is split into two halves, one half is used for transmission the other for reception. Only 8-bit fields can be fetched from memory so two tests are required for 16-bit systems to verify integrity of the entire data path. During loopback the maximum latency to obtain access to the buffer memory is 2.0  $\mu$ s. Systems that wish to use the loopback test yet do not meet this latency can limit the loopback packet to 7 bytes without experiencing underflow. Only the last 8 bytes of the loopback packet are retained in the FIFO. The last 8 bytes can be read through the FIFO register which will advance through the FIFO to allow reading the receive packet sequentially.

Destination Address	= 6 bytes Station Physical Address
Source Address	= 6 bytes Station Physical Address
Length	2 bytes
Data	= 46 to 1500 bytes
CRC	Appended by DP8307 Controller if CRC = 0 in TCR

When in word-wide mode with Byte Order Select set, the loopback packet must be assembled in the even byte locations as shown below. (The loopback only operated with byte wide transfers.)

LS Byte (D6-D15)	MS Byte (D0-D7)
	Destination
	Source
	Length
	Data
	CRS

WTS = 1      BOS = 1      (DCR Bits)

When in word-wide mode with Byte Order Select low, the following format must be used for the loopback packet.

MS Byte (D8-D15)	LS Byte (D0-D7)
Destination	
Source	
Length	
Data	
CRS	

WTS = 1      BOS = 0      (DCR Bits)

**Note:** When using loopback in word mode 2n bytes must be programmed in the TBCR0, 1. When n = actual number of bytes assembled in even or odd location.

To initiate a loopback the user first assembles the loopback packet then selects the type of loopback using the Transmit Configuration register bits LB0, LB1. The transmit configuration register must also be set to enable or disable CRC generation during transmission. The user then issues a normal transmit command to send the packet. During loopback the receiver checks for an address match and if CRC bit in the TCR is set, the receiver will also check the CRC. The last 8 bytes of the loopback packet are buffered and can read out of the FIFO using FIFO read port.

#### Loopback Modes

**MODE 1:** Loopback Through the DP83907 Controller Module (LB1 = 0, LB0 = 1): If this loopback is used, The DP83907 Module's serializer is connected to the deserializer.

**MODE 2:** Loopback Through the ENDEC Module (LB1 = 1, LB0 = 0): If the loopback is to be performed through the SNI, the DP83907 provides a control (LPBK) that forces the ENDEC module to loopback all signals.

**MODE 3:** Loopback to the external coax interface or twisted pair interface module (LB1 = 1, LB0 = 1).

Packets can be transmitted to the cable in loopback mode to check all of the transmit and receive paths and the cable itself. If, in twisted pair mode, there is a link fail the transmitter will be disabled which could give misleading results in Mode 3. The link integrity should be checked, by reading Configuration Register B, before this test.

#### Reading the Loopback Packet

The last eight bytes of a received packet can be examined by 8 consecutive reads of the FIFO register. The FIFO pointer is increment after the rising edge of the CPU's read strobe by internally synchronizing and advancing the pointer. This may take up to four bus clock cycles, if the pointer has not been incremented by the time the CPU reads the FIFO register again, the DP83907 will insert wait states.

**Note:** The FIFO may only be read during Loopback. Reading the FIFO at any other time will cause the DP83907 to malfunction.

#### Alignment of the Received Packet in the FIFO

Reception of the packet in the FIFO begins at location zero, after the FIFO pointer reaches the last location in the FIFO, the pointer wraps to the top of the FIFO overwriting the previously received data. This process continued until the last byte is received. The DP83907 then appends the received byte count in the next two locations of the FIFO. The contents of the Upper Byte Count are also copied to the next FIFO location. The number of bytes used in the loopback packet determined the alignment of the packet in the FIFO. The alignment for a 64-byte packet is shown below.

FIFO Location	FIFO Contents	
0	Lower Byte Count	→ First Byte Read
1	Upper Byte Count	→ Second Byte Read
2	Upper Byte Count	•
3	Last Byte	•
4	CRC1	•
5	CRC2	•
6	CRC3	•
7	CRC4	→ Last Byte Read



## 6.0 Operation of DP83907 (Continued)

For the following alignment in the FIFO the packet length should be  $(N \times 8) + 5$  Bytes. Note that if the CRC bit in the TCR is set, CRC will not be appended by the transmitter. If the CRC is appended by the transmitter, the 1st four bytes, bytes N-3 to N, correspond to the CRC.

FIFO Location	FIFO Contents		
0	Byte N-4	→	First Byte Read
1	Byte N-3 (CRC1)	→	Second Byte Read
2	Byte N-2 (CRC2)		•
3	Byte N-1 (CRC3)		•
4	Byte N (CRC4)		•
5	Lower Byte Count		•
6	Upper Byte Count		•
7	Upper Byte Count	→	Last Byte Read

### Loopback Tests

Loopback capabilities are provided to allow certain tests to be performed to validate operation of the DP83907 prior to transmitting and receiving packets on a live network. Typically these tests may be performed during power up of a node. The diagnostic provides support to verify the following:

- 1) Verify integrity of data path. Received data is checked against transmitted data.
- 2) Verify CRC logic's capability to generate good CRC on transmit, verify CRC on receive (good or bad CRC).
- 3) Verify that the Address Recognition Logic can
  - a) Recognize address match packets
  - b) Reject packets that fail to match an address

### Loopback Operation in the DP83907 Controller

Loopback is a modified form of transmission using only half of the FIFO. This places certain restrictions on the use of loopback testing. When loopback mode is selected in the TCR, the FIFO is split. A packet should be assembled in memory with programming of TPSR and TBCR0, TBCR1 registers. When the transmit command is issued the following operations occur:

### TRANSMITTER ACTIONS

- 1) Data is transferred from memory by the DMA until the FIFO is filled. For each transfer TBCR0 and TBCR1 are decremented. (Subsequent burst transfers are initiated when the number of bytes in the FIFO drops below the programmed threshold.)
- 2) The DP83907 generates 56 bits of preamble followed by an 8-bit synch pattern.
- 3) Data transferred from FIFO to serializer.
- 4) If CRC = 1 in TCR, no CRC calculated by DP83907, the last byte transmitted is the last byte from the FIFO (Allows software CRC to be appended). If CRC = 0, DP83907 calculates and appends four bytes of CRC.
- 5) At end of Transmission PTX bit set in ISR.

### RECEIVER ACTIONS

- 1) Wait for synch, all preamble stripped.
- 2) Store packet in FIFO, increment receive byte count for each incoming byte.
- 3) If CRC = 0 in TRC, receiver checks incoming packet for CRC errors. If CRC = 1 in TCR, receiver does not check CRC errors, CRC error bit always set in RSR (for address matching packets).
- 4) At end of receive, receive byte count written into FIFO, receive status register is updated. The PRX bit is typically set in the RSR even if the address does not match. If CRC errors are forced, the packet must match the address filters in order for the CRC error bit in the RS to be set.

### EXAMPLES

The following examples show what results can be expected from a properly operating DP83907 during loopback. The restrictions and results of each type of loopback are listed for reference. The loopback tests are divided into two sets of tests. One to verify the data path, CRC generation and byte count through all three paths. The second set of tests uses internal loopback to verify the receiver's CRC checking and address recognition. For all of the tests the DCR was programmed to 40H.

Path	TCR	RCR	TSR	RSR	ISR
DP83907 Internal	02	00	53 (Note 1)	02 (Note 2)	02 (Note 3)

**Note 1:** Since carrier sense and collision detect are generated in the ENDEC module. They are blocked during internal loopback, carrier and CD heartbeat are not seen and the CRS and CDH bits are set.

**Note 2:** CRC errors are always indicated by receiver if CRC is appended by the transmitter.

**Note 3:** Only the PTX bit in the ISR is set, the PRX bit is only set if status is written to memory. In loopback this action does not occur and the PRX bit remains 0 for all loopback modes.

**Note 4:** All values are hex.

Path	TCR	RCR	TSR	RSR	ISR
DP83907 Internal	04	00	43 (Note 1)	02	02

**Note 1:** CDH is set, CRS is not set since it is generated by the external encoder/decoder.

Path	TCR	RCR	TSR	RSR	ISR
DP83907 Internal	06	00	03 (Note 1)	02	02 (Note 2)

**Note 1:** CDH and CRS should not be set. The TSR however, could also contain 01H, 03H, 07H and a variety of other values depending on whether collisions were encountered or the packet was deferred.

**Note 2:** Will contain 08H if packet is not transmittable.

**Note 3:** During external loopback the DP83907 Controller is now exposed to network traffic, it is therefore possible for the contents of both the Receive portion of the FIFO and the RSR to be corrupted by any other packet on the network. Thus in a live network the contents of the FIFO and RSR should not be depended on. The DP83907 will still abide by the standard CSMA/CD protocol in external loopback mode. (i.e. The network will not be disturbed by the loopback packet).

**Note 4:** All values are hex.

## 6.0 Operation of DP83907 (Continued)

### CRC and Address Recognition

The next three tests exercise the address recognition logic and CRC. These tests should be performed using internal loopback only so that the DP83907 is isolated from interference from the network. These tests also require the capability to generate CRC in software.

The address recognition logic cannot be directly tested. The CRC and FAE bits in the RSR are only set if the address if the packet matches the address filters. If errors are expected to be set and they are not set, the packet has been rejected on the basis of an address mismatch. The following sequence of packets will test the address recognition logic. The DCR should be set to 40H, the TCR should be set to 03H with a software generated CRC.

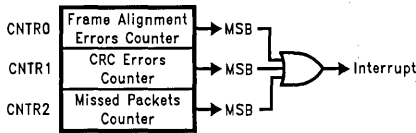
Packet Contents		Results	
Test	Address	CRC	RSR
Test A	Matching	Good	01 (Note 1)
Test B	Matching	Bad	02 (Note 2)
Test C	Non-Matching	Bad	01

**Note 1:** Status will read 21H if multicast address used.

**Note 2:** Status will read 22H if multicast address used.

**Note 3:** In test A, the RSR is set up. In test B the address is found to match since the CRC is flagged as bad. Test C proves that the address recognition logic can distinguish a bad address and does not notify the RSR of the bad CRC. The receiving CRC is proven to work in test A and test B.

**Note 4:** All values are hex.



TL/F/12082-31

FIGURE 36. Tally Counters

### Network Management Functions

Network management capabilities are required for maintenance and planning of a local area network. The DP83907 supports the minimum requirement for network management in hardware, the remaining requirements can be met with software. Software alone can not track during reception of packets: CRC errors, Frame Alignment errors, and missed packets, *Figure 36*.

Since errored packets can be rejected, the status associated with these packets is lost unless the CPU can access the Receive Status Register before the next packer arrives. In situations where another packet arrives very quickly, the CPU may have no opportunity to do this. The DP83907 Controller counts the number of packets with CRC errors and Frame Alignment errors. 8-bit counters have been selected to reduce overhead. The counters will generate interrupts whenever their MSBs are set so that a software routine can accumulate the network statistics and reset the counter before overflow occurs. The counters are sticky so that when they reach a count of 192 (C0H) counting is halted. An additional counter is provided to count the number of packets the DP83907 misses due to buffer overflow or being off-line. The structure of the counters is shown in *Figure 36*.

Additional information required for network management is available in the Receive and Transmit Status Registers. Transmit status is available after each transmission for information regarding events during transmission.

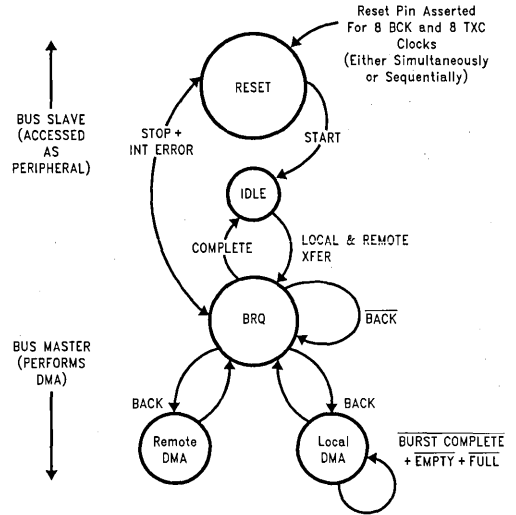
Typically, the following statistics might be gathered in software:

- Traffic:
  - Frames Sent OK
  - Frames Received OK
  - Multicast Frames Received
  - Packets Lost Due to Lack of Resources
  - Retries/Packet
- Errors:
  - CRC Errors
  - Alignment Errors
  - Excessive Collisions
  - Packet with Length Errors
  - Heartbeat Failure

### 6.6 MEMORY ARBITRATION AND BUS OPERATION

The DP83907 will always operate as a slave device on it's peripheral interface to the ISA bus. However on the memory bus, the DP83907 Controller operates in three possible modes:

1. Bus Master of Local Packet Buffer RAM
2. Bus Slave when accessed by the CPU via the Bus Interface
3. Idle, when no activity is occurring.



TL/F/12082-32

FIGURE 37. DP8390 Core Bus States

Upon power-up the DP83907 is in an indeterminate state. After receiving a hardware reset the DP83907 is a bus slave in the Reset State, the receiver and transmitter are both disabled in this state. The reset state can be re-entered under four conditions, soft reset (Stop Command), register reset (reset port), hard reset (RESET input) or an error that shuts down the receiver of transmitter (FIFO underflow or overflow, receive buffer ring overflow).

After initialization of registers, the DP83907 Controller is issued a Start command and the DP83907 enters Idle state. Until the DMA is required the DP83907 remains in idle state.

## 6.0 Operation of DP83907 (Continued)

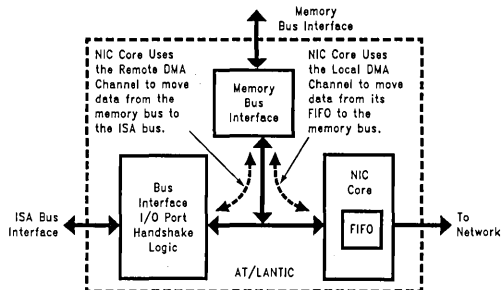
The idle state is exited and the DP83907 will drive the local memory bus when a request from the FIFO in the DP8390 (NIC) core causes the memory bus interface logic to issue a read or write operation, such as when the DP83907 is transmitting or receiving data.

The NIC Core's Remote DMA also requests access from the memory bus. When software programs an I/O data transfer between the CPU and the buffer RAM, the Remote DMA controls this request.

All Local DMA transfers are burst transfers, the DMA will transfer an exact burst of bytes programmed in the Data Configuration Register (DCR) then relinquish the memory bus. If there are remaining bytes in the FIFO the next burst will not be initiated until the FIFO threshold is exceeded.

### I/O Mode Operation

In I/O mode the DP83907 transfers data to and from the packet buffer RAM by utilizing the Remote DMA logic which is programmed by the main system CPU to transfer data through the DP83907's internal data



TL/F/12082-33

**FIGURE 38. I/O Operation: All Data Transfers and Arbitration is Controlled by the NIC Core**

### INTERLEAVED LOCAL/REMOTE OPERATION

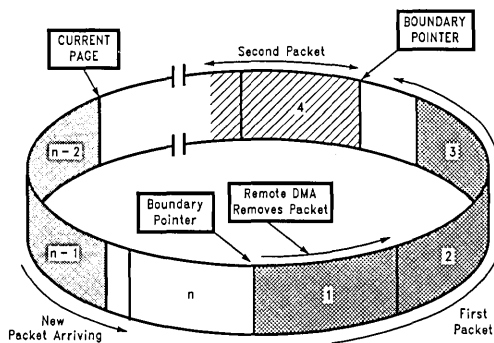
When in I/O mode the remote DMA is used to transfer data to/from the main system. If a remote DMA transfer is initiated or in progress when a packet is being received or transmitted, the Remote DMA transfers will be interrupted for higher priority Local DMA transfers. When the Local DMA transfer is completed the Remote DMA will re-arbitrate for the bus and continue its transfers.

If the FIFO requires service while a remote DMA is in progress the Local DMA burst is appended to the Remote Transfer. When switching from a local transfer to a remote transfer there is a break to allow the CPU to fairly contend for the bus.

### REMOTE DMA BI-DIRECTIONAL PORT

The Remote DMA transfers data between the local buffer memory and the internal bi-directional port (memory to I/O transfer).

This transfer is arbitrated on a transfer by transfer basis versus the burst transfer mode used for Local DMA transfers. This bi-directional port is integrated onto the DP83907, and is read/written by the host. All transfers through this port are asynchronous. At any one time transfers are limited to one direction, either form the port to local buffer memory (Remote Write) or form local buffer memory to the port (Remote Read).



TL/F/12082-34

**FIGURE 39. 1st Received Packet Removed by Remote DMA**

### I/O MODE REMOVING PACKETS FROM RING

Network activity is isolated on a local bus, where the DP83907's local DMA channel performs burst transfers between the buffer memory and the DP83907's FIFO. The Remote DMA transfers data between the buffer memory and the host memory via the internal bi-directional I/O port. The Remote DMA provides local addressing capability and is used as a slave DMA by the host. The host system reads the I/O port to transfer data between the system and I/O port. The DP83907 allows Local and Remote DMA operations to be interleaved.

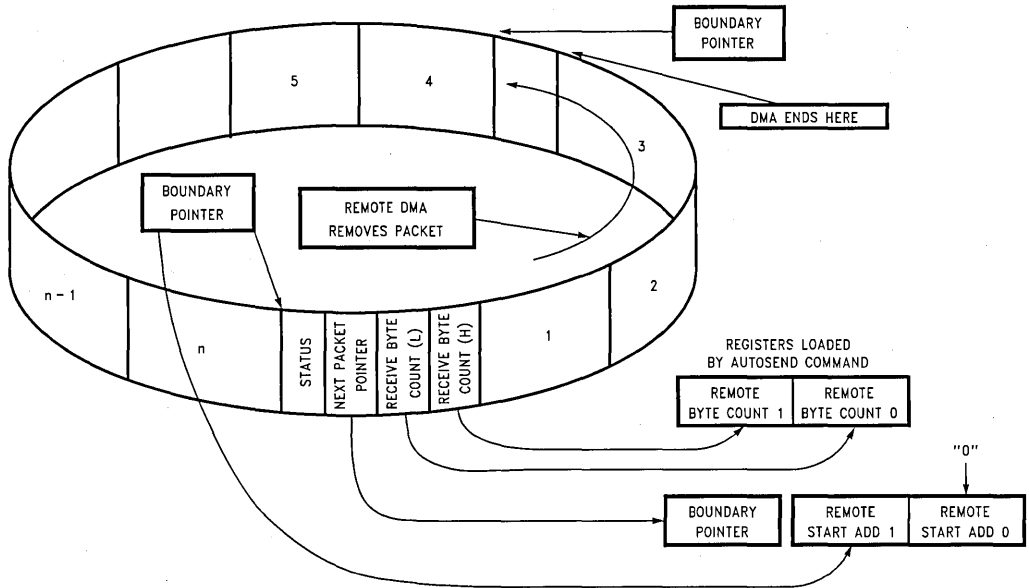
Packets are removed from the ring using the Remote DMA. When using the Remote DMA the Send Packet command can be used. This programs the Remote DMA to automatically remove the received packet pointed to by the Boundary Pointer. At the end of the transfer, the DP83907 moves the Boundary Pointer, freeing additional buffers for reception. The Boundary Pointer can also be moved manually by programming the Boundary Register. Care should be taken to keep the Boundary Pointer at least one buffer behind the Current Page Pointer.

Data transfer by the Remote DMA to the integrated I/O data port is dependent on whether the DP83907 is set into 8 bit mode or 16 bit mode. In 8 bit mode all transfers are 8 bits (1 byte) wide. When in 16 bit mode all transfers are 16 bits (1 word) wide. The data width is selected by setting the WTS bit in the data Configuration Register and setting the DWID pin for the proper mode.

The following is a suggested method for maintaining the Receive Buffer Ring pointers:

1. At initialization, set up a software variable (`next_pkt`) to indicate where the next packet will be read. At the beginning of each Remote Read DMA operation, the value of `next_pkt` will be loaded into RSAR0 and RSAR1.
2. When initializing the DP83907 Controller set:
  - BNDRY = PSTART
  - CURR = PSTART + 1
  - `next_pkt` = PSTART + 1
3. After a packet is DMAed from the Receive Buffer Ring, the Next Page Pointer (second byte in the DP83907 Controller buffer header) is used to update BNDRY and `next_pkt`.

6.0 Operation of DP83907 (Continued)



TL/F/12082-35

FIGURE 40. Remote DMA Auto Initialization from Buffer Ring

next\_pkt = Next Page Pointer

BNDRY = Next Page Pointer - 1

If BNDRY < PSTART then BNDRY = PSTOP - 1

Note the size of the Receive Buffer Ring is reduced by one 256-byte buffer, this will not, however impede the operation of the DP83907. The advantage of this scheme is that it easily differentiates between buffer full and buffer empty: it is full if BNDRY = CURR; empty when BNDRY = CURR-1. If, in I/O mode, send packet is used to empty the buffer ring this scheme cannot be used. BNDRY must be initialized equal to CURR, or the first executed send packet will not return data from the received packet, which will be written at CURR. The Overwrite Warning bit of the Interrupt Status Register must be used in this mode to differentiate between buffer full and buffer empty.

**I/O REMOTE DMA COMMANDS**

The Remote DMA channel is used in the I/O Mode to both assemble packets for transmission, and to remove received packets from the Receive Buffer Ring. It may also be used for moving blocks of data or commands between host memory and local buffer memory. (In Shared Memory Mode, the Remote DMA should be disabled, and not used. Packet transfer to/from the system is accomplished by normal CPU read/write operations.)

There are three modes of Remote DMA operation: Remote Write, Remote Read, or Send Packet.

Two register pairs are used to control the Remote DMA, a Remote Start Address (RSAR0, RSAR1) and a Remote Byte Count (RBCRO, RBCR1) register pair. The Start Address Register pair points to the beginning of the block to be moved while the Byte Count Register pair is used to indicate the number of bytes to be transferred. Full handshake logic is provided to move data between local buffer memory and a bi-directional I/O port.

**Remote Write:** A Remote Write transfer is used to move a block of data from the host into local buffer memory. The Remote DMA will read data from the I/O port and sequentially write it to local buffer memory beginning at the Remote Start Address. The DMA Address will be incremented and the Byte Counter will be decremented after each transfer. The DMA is terminated when the Remote Byte Count Register reaches a count of zero.

**Remote Read:** A Remote Read transfer is used to move a block of data from local buffer memory to the host. The Remote DMA will sequentially read data from the local buffer memory, beginning at the Remote Start Address, and write data to the I/O port.

## 6.0 Operation of DP83907 (Continued)

The DMA Address will be incremented and the Byte Counter will be decremented after each transfer. The DMA is terminated when the Remote Byte Count Register reaches zero.

**Send Packet Command:** The Remote DMA channel can be automatically initialized to transfer a single packet from the Receive Buffer Ring. The CPU begins this transfer by issuing a "Send Packet" Command. The DMA will be initialized to the value of the Boundary Pointer Register and the Remote Byte Count Register pair (RBCR0, RBCR1) will be initialized to the value of the Receive Byte Count fields found in the Buffer Header of each packet. After the data is transferred, the Boundary Pointer is advanced to allow the buffers to be used for new receive packets. The Remote Read will terminate when the Byte Count equals zero. The Remote DMA is then prepared to read the next packet from the Receive Buffer Ring. If the DMA pointer crosses the Page Stop Register, it is reset to the Page Start Address. This allows the Remote DMA to remove packets that have wrapped around to the top of the Receive Buffer Ring.

**Note 1:** In order for the DP83907 Controller to correctly execute the Send Packet Command, the upper Remote Byte Count Register (RBCR1) must first be loaded with 0FH.

**Note 2:** The Send Packet command cannot be used with 680x0 type processors.

### I/O READ TIMING

- 1) The DMA reads word from local buffer memory and writes the word into the internal latch, increments the DMA address and decrements the byte count (RBCR0,1).
- 2) Internally a request line is asserted to enable the system to read the port. If the system reads this port before the data has been written, then the system is sent a wait signal to wait until the data has been written to the port. Once written the systems read is allowed to complete.
- 3) The system reads the port, the read strobe for the port is used as an acknowledgement to the Remote DMA and it goes back to step 1.

Steps 1–3 are repeated until the remote DMA is complete. (i.e., the byte count has gone to zero.)

Note that in order for the Remote DMA to transfer a word from memory to the latch, it must arbitrate access to the local buffer RAM. After each word is transferred to the internal latch, access to the RAM is relinquished. If a Local DMA is in progress, the Remote DMA is held off until the local DMA is complete.

### I/O WRITE TIMING

A Remote Write operation transfers data from the I/O port to the local buffer RAM. The system transfers a byte-word to the latch via IOWR. This write strobe is detected by the DP83907 and the byte/word is transferred to local buffer memory. The Remote DMA holds off further transfers into the latch until the current byte/word has been transferred from the latch.

- 1) DP83907 awaits data to be written by the system. System writes byte/word into latch.
- 2) Remote DMA reads contents of port and writes byte/word to local buffer memory, increments address and decrements byte count (RBCR0,1).
- 3) Go back to step 1.  
Steps 1–3 are repeated until the remote DMA is complete.

### 6.7 FUNCTIONAL BUS TIMING

This section describes the bus cycles that the DP83907 performs. These timings can be subdivided into 3 basic categories:

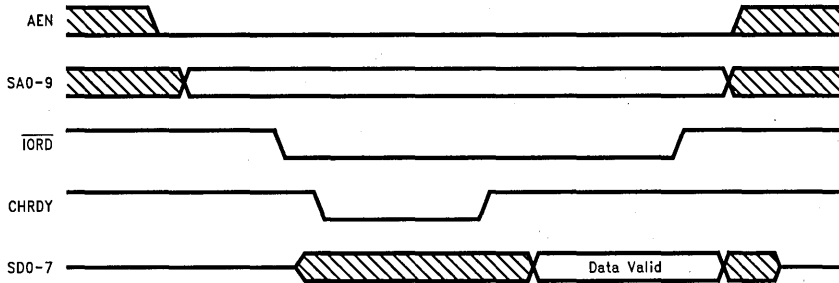
1. ISA I/O Access: There are register accesses in both modes, and I/O data accesses in I/O mode.
2. Shared RAM ISA Accesses: These are the timing for the ISA bus accesses through the DP83907 to the memory bus and buffer RAM.
3. Boot PROM ISA Accesses: These are the timing for the ISA bus accesses through the DP83907 to the memory bus and boot PROM.
4. Local and I/O RAM Accesses: This is the timing of the Local DMA, accesses from the NIC Core FIFO to the RAM, and the Remote DMA accesses to the RAM over the memory bus.

### ISA Bus I/O Accesses

The DP83907 is designed to directly interface to the ISA bus (PC AT backplane bus). The CPU can read or write any internal registers. All register accesses are byte wide. The functional timing for DP83907 accesses are shown in the following pages.

## 6.0 Operation of DP83907 (Continued)

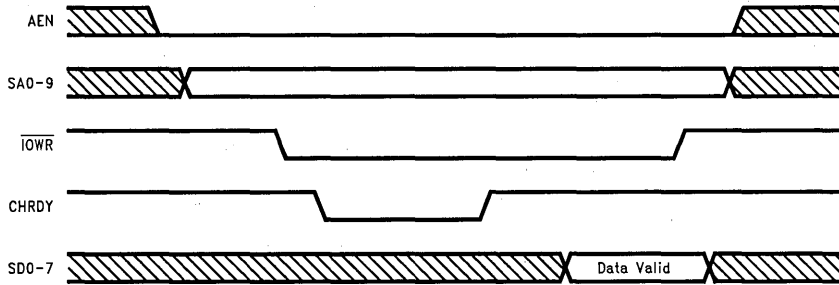
### 8-Bit I/O Slave Read



TL/F/12082-36

This is the type of cycle used to read from a register or, in 8-bit mode, from a data transfer port. These accesses are entirely asynchronous, with the DP83907 responding when it decodes the correct address on SA0-9 and an  $\overline{\text{IORD}}$ . If AEN is high the cycle will be ignored. CHRDY is deasserted if the DP83907 is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted. The data will always appear on SD0-7.

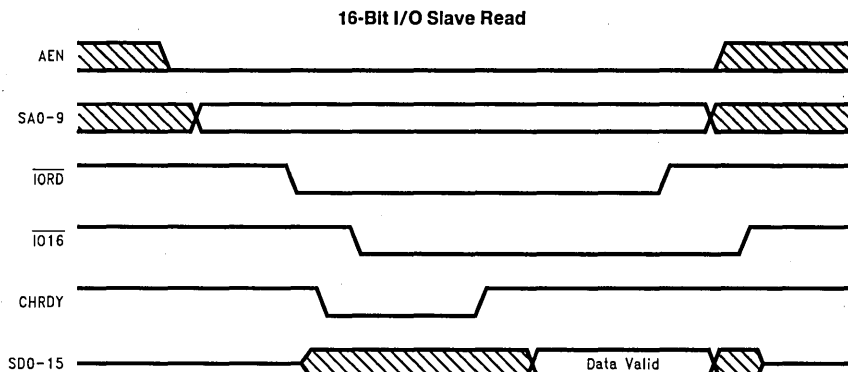
### 8-Bit I/O Slave Write



TL/F/12032-37

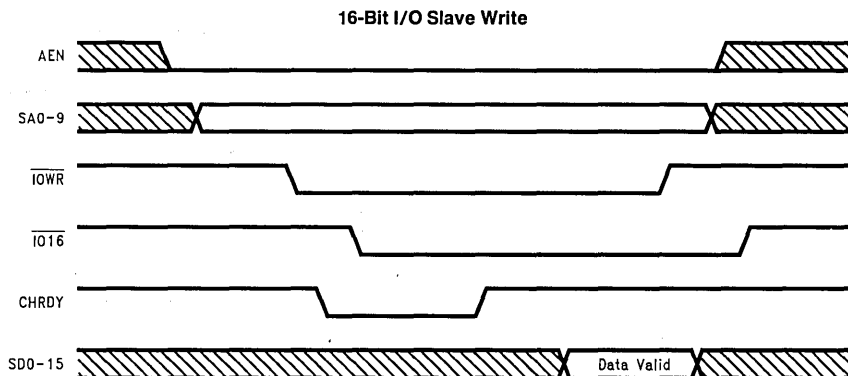
This is the type of cycle used to write to a register or, in 8-bit mode, to a data transfer port. These accesses are entirely asynchronous, with the DP83907 responding when it decodes the correct address on SA0-9 and an  $\overline{\text{IOWR}}$ . If AEN is high the cycle will be ignored. CHRDY is deasserted if the DP83907 is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted. The data will always be taken from SD0-7.

## 6.0 Operation of DP83907 (Continued)



TL/F/12082-38

This is the type of cycle used to read from a data transfer port in 16-bit mode. These accesses are entirely asynchronous, with the DP83907 responding when it decodes the correct address on SA0-9 and an  $\overline{\text{IORD}}$ . If AEN is high the cycle will be ignored. CHRDY is deasserted if the DP83907 is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted.  $\overline{\text{IO16}}$  is generated, when an address within the DP83907's data transfer port is decoded, to indicate to the system that this is a 16-bit transfer. If the IO16CON bit in Configuration Register B is low then it will be a straight decode of the SA0-9 lines. If that bit is high the  $\overline{\text{IO16}}$  output will be generated after  $\overline{\text{IORD}}$  goes active, to indicate that this is a 16-bit transfer, and the address should be even, SA0 low. The data will appear on SD0-15.



TL/F/12082-39

This is the type of cycle used to write to a data transfer port in 16-bit mode. These accesses are entirely asynchronous, with the DP83907 responding when it decodes the correct address on SA0-9 and an  $\overline{\text{IOWR}}$ . If AEN is high the cycle will be ignored. CHRDY is deasserted if the DP83907 is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted.  $\overline{\text{IO16}}$  is generated, when an address within the DP83907's data transfer port is decoded, to indicate to the system that this is a 16-bit transfer. If the IO16CON bit in Configuration Register B is low then it will be a straight decode of the SA0-9 lines. If that bit is high the  $\overline{\text{IO16}}$  output will be generated after  $\overline{\text{IOWR}}$  goes active, to indicate that this is a 16-bit transfer, and the address should be even, SA0 low. The data will be taken from SD0-15.

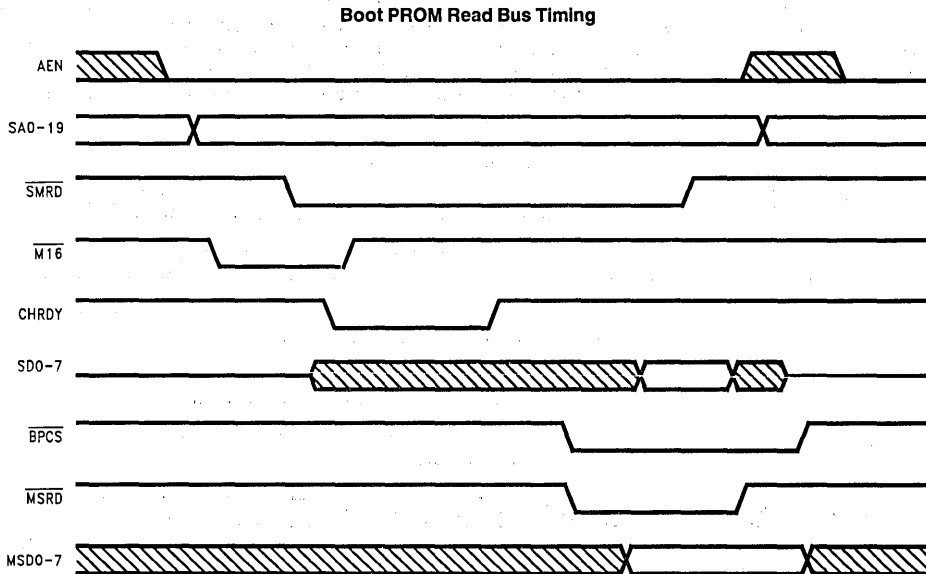
## 6.0 Operation of DP83907 (Continued)

### 16-Bit I/O Cycle with IO16 Fix

Some Chips & Technologies and VLSI Technologies PC-AT chip sets have timing requirements in 16-bit I/O cycles that cannot be achieved by the default DP83907 cycle, described on the previous page. When that cycle is executed with these chip sets, the system does not recognize the CHRDY signal and does not insert wait states. The system executes a standard cycle and deasserts IORD or IOW even if CHRDY is still deasserted. The DP83907 recognizes if this situation has occurred, asserts CHRDY and sets a bus error bit in Configuration Register B to flag this error. Thus the user can test any new system to see if this error occurs and then take some remedial action. The DP83907 supports a fix which can be selected by software, by writing to Configuration Register B.

This fix is enabled by setting the IO16 bit of Configuration Register B. In normal operation any time a valid address exists on SA0-9 IO16 is generated. Delaying IO16 until after the IORD or IOW can cure the problem on non-compliant machines. The theory is that the system is fooled into thinking an 8-bit peripheral is responding, since IO16 is not generated for the valid address, and accepts 8-bit I/O cycle timings for CHRDY. It then rechecks IO16 after the IORD or IOW strobe and correctly determines it is a 16-bit peripheral. If a system did not recheck IO16 it would generate 2, 8-bit cycles instead of 1, 16-bit cycle. The DP83907 would interpret each 8-bit access as a 16-bit transfer and decrement it's DMA byte count by 2. Eventually the system would attempt to access the data transfer port when the DP83907 had finished transferring data and CHRDY would be deasserted indefinitely. To prevent misoperation this fix should only be implemented on systems that require it.

### ISA BUS BOOT PROM ACCESS TIMING

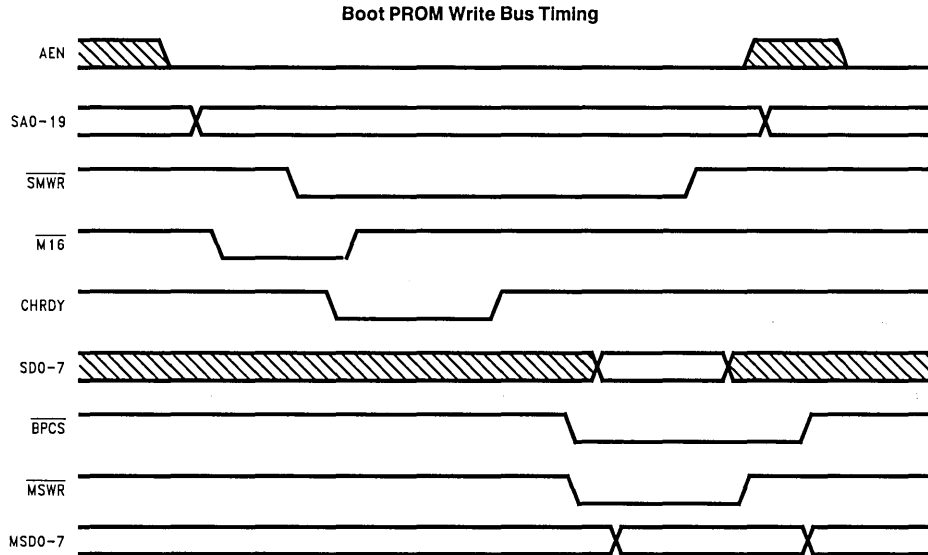


TL/F/12082-40

This is the type of cycle used to read the boot PROM. These accesses are entirely asynchronous, with the DP83907 responding when it decodes the correct address on SA0-19 and a SMRD. If AEN is high the cycle will be ignored. CHRDY is deasserted if the DP83907 is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted. The data will be driven from MSD0-7 onto SD0-7.



## 6.0 Operation of DP83907 (Continued)

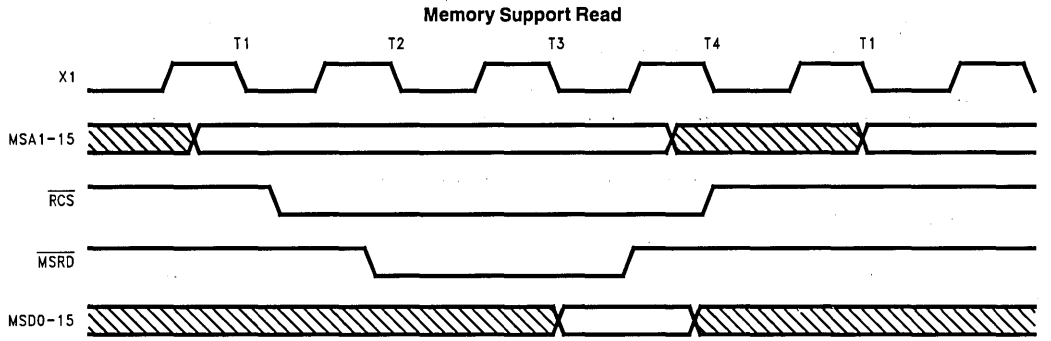


TL/F/12082-41

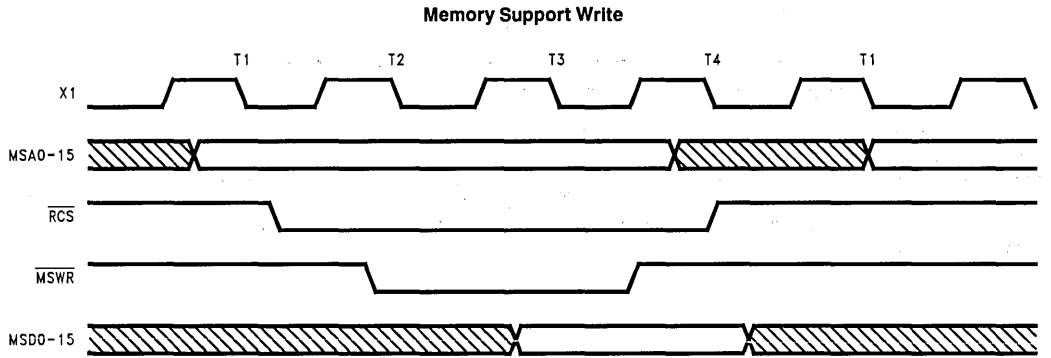
This is the type of cycle used to write to the boot PROM. These accesses are entirely asynchronous, with the DP83907 responding when it decodes the correct address on SA0-19 and a  $\overline{SMWR}$ . If AEN is high the cycle will be ignored. CHRDY is deasserted if the DP83907 is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted. The data will normally be taken from SD0-7. However, if  $\overline{M16}$  is generated and the access is to an odd address the data will be taken from SD8-15. The data will always be driven onto MSD0-7. The BPWR bit of Configuration Register B must be high to allow write cycles to the boot PROM.

## 6.0 Operation of DP83907 (Continued)

### RAM ACCESS TIMING



This is a memory read cycle executed by the DP83907's internal DMA. This is used to either load the data transfer port, during a Remote Read in I/O mode, or to load the FIFO, for a transmission on the network, in both modes. This transfer is synchronized to X1.



This is a memory write cycle executed by the DP83907's internal DMA. This is used to either write from the data transfer port, during a Remote Write in I/O mode, or to empty the FIFO, during a reception from the network, in both modes. This transfer is synchronized to X1.

## 7.0 In-Circuit Test (ICT) Description

The test register is made available only when the test bit (D7 of cnfg. A) is set at the following I/O address:

I/O mode: Base + 1FHex.

The test register contains the following bits. Note that when the test bit is low the test register is held set except for D4 and D0 which is reset.

TABLE I. Test Modes (test pin = 1)

	Test Mode	D7	D6	D5	D4	D3	D2	D1	D0	SA<9>
1	V <sub>IH</sub> /V <sub>IL</sub>	1	1	1	1	1	1	1	0	X
2	V <sub>OH</sub> /V <sub>OL</sub>	1	1	1	0	1	0	1	0	1

V<sub>IL</sub>/V<sub>IH</sub> test mode when high will AND all inputs (except RESET pin) and present outcome on IRQ<3>. To enable this st-nc mode must NOT be entered.

V<sub>OH</sub>/V<sub>OL</sub> test mode is active in st-nc mode if SA<9> = 1. Then all o/p's switch with SA<4> as per Table 2.

V<sub>OH</sub>/V<sub>OL</sub> mode output states (SA<9> = 1)

TABLE II

Output	SA<4> = 0	SA<4> = 1
MSA<13:1>	1555 Hex	0AAA Hex
MSD<15:0>	AAAA Hex	5555 Hex
SD<15:0>	AAAA Hex	5555 Hex
IRQ<15, 12, 11, 10, 9, 5, 4, 3>	A2 Hex	5D Hex
MSWR	0	1
MSRD	1	0
RCS	0	1
BPCS	0	1
EECS	1	0
IO16	0	0
CHRDY (AEN, AORD Must be Low)	0	0
GDLNK, ACT, COL	0, 1, 0	1, 0, 1
THIN	1	0

## DP83907 Preliminary Specification Design Guide

This document provides information on National Semiconductor's DP83907 target design parametric specifications. These specifications are provided as design targets for the DP83907. These parameters should be used to undergo systems design activities, and are not provided as guaranteed parametric specifications. At this time these specifications are **not** tested, and are subject to change based upon National's product characterization and yield analysis.

### Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage ( $V_{CC}$ )	-0.5V to +7.0V
Storage Temperature ( $T_{STG}$ )	-65°C to +150°C
Package Power Dissipation ( $P_D$ )	900 mW
Lead Temperature ( $T_L$ ) (Soldering, 10 seconds)	260°C

### Operating Conditions

	Min	Max	Units
Supply Voltage ( $V_{CC}$ )	4.75	5.25	V
Operating Temperature ( $T_A$ )	0	+70	°C
ESD Tolerance:	2.0		kV
$C_{ZAP} = 100 \text{ pF}$ , $R_{ZAP} = 1.5 \text{ k}\Omega$			

### Preliminary DC Specifications

Symbol	Description	Conditions	Min	Max	Units
<b>SUPPLY CURRENT</b>					
$I_{CC}$	Average Active (Transmitting and Receiving) Supply Current	$X1 = 20 \text{ MHz Clock}$ $V_{IN} = \text{Switching}$		150	mA
$I_{CCIDLE}$	Average Idle Supply Current	$X1 = 20 \text{ MHz Clock}$ $V_{IN} = V_{CC} \text{ or GND}$		120	mA
<b>TTL INPUTS</b>					
$V_{IL}$	Maximum Low Level Input Voltage			0.8	V
$V_{IH}$	Minimum High Level Input Voltage		2.0		V
$I_{IN}$	Input Current	$V_I = V_{CC} \text{ or GND}$	-10	+10	$\mu\text{A}$
<b>3ST TRI-STATE HIGH DRIVE I/O</b>					
$V_{OH}$	Minimum High Level Output Voltage	$I_{OH} = -3 \text{ mA}$	2.4		V
$V_{OL}$	Maximum Low Level Output Voltage	$I_{OL} = 24 \text{ mA}$		0.5	V
$V_{IL}$	Maximum Low Level Input Voltage			0.8	V
$V_{IH}$	Minimum High Level Input Voltage		2.0		V
$I_{IN}$	Input Current	$V_I = V_{CC} \text{ or GND}$	-10	+10	$\mu\text{A}$
$I_{OZ}$	Maximum TRI-STATE Output Leakage Current	$V_{OUT} = V_{CC} \text{ or GND}$	-10	+10	$\mu\text{A}$
<b>MOS INPUTS, OUTPUTS AND I/O</b>					
$V_{OH}$	Minimum High Level Output Voltage	$I_{OH} = -20 \mu\text{A}$	$V_{CC} - 0.1$		V
$V_{OL}$	Maximum Low Level Output Voltage	$I_{OL} = 20 \mu\text{A}$		0.1	V
$V_{IL}$	Maximum Low Level Input Voltage			0.8	V
$V_{IH}$	Minimum High Level Input Voltage		2.0		V
$I_{IN}$	Input Current	$V_I = V_{CC} \text{ or GND}$	-10	+10	$\mu\text{A}$
$I_{IN1}$	Input Current MSA9-13, MSD0-2	$V_I = V_{CC} \text{ or GND}$	-10	+10	$\mu\text{A}$
$I_{IN2}$	Input Current MSD3-15, MSA1-8	$V_I = V_{CC} \text{ or GND}$ RESET = Active	-10	+10	$\mu\text{A}$
$I_{OZ}$	Maximum TRI-STATE Output Leakage Current	$V_{OUT} = V_{CC} \text{ or GND}$	-10	+10	$\mu\text{A}$
<b>OCH OPEN COLLECTOR HIGH DRIVE OUTPUT</b>					
$V_{OL}$	Maximum Low Level Output Voltage	$I_{OL} = 24 \text{ mA}$		0.5	V
<b>LED DRIVER OUTPUT</b>					
$V_{OL}$	Maximum Low Level Output Voltage	$I_{OL} = 16 \text{ mA}$		0.5	V

## Preliminary DC Specifications (Continued)

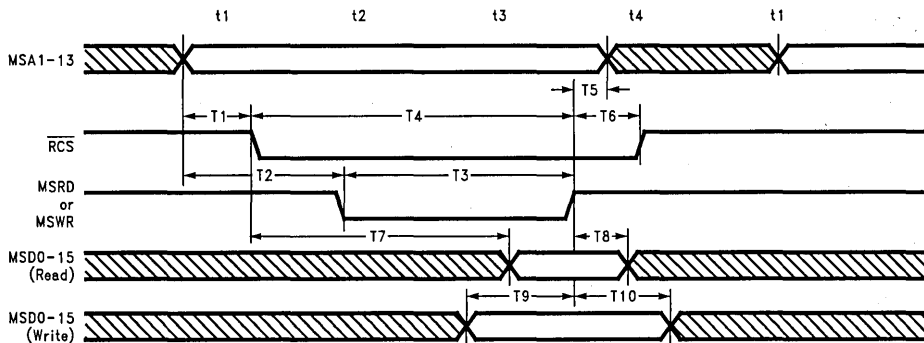
Symbol	Description	Conditions	Min	Max	Units
<b>OSCILLATOR PINS (X1 AND X2)</b>					
$V_{IH}$	X1 Input High Voltage	X1 is Connected to an Oscillator	2.0		V
$V_{IL}$	X1 Input Low Voltage	X1 is Connected to an Oscillator		0.8	V
$I_{OSC}$	X1 Input Current	X1 is Connected to an Oscillator $V_{IN} = V_{CC}$ or GND		1	mA
<b>TPI</b>					
$R_{TOL}$	$TXO_d \pm$ , $TXO \pm$ Low Level Output Resistance	$I_{OL} = 25$ mA		15	$\Omega$
$R_{TOH}$	$TXO_d \pm$ , $TXO \pm$ High Level Output Resistance	$I_{OH} = -25$ mA		15	$\Omega$
$V_{SRON1}$	Receive Threshold Turn-On Voltage 10BASE-T Mode		$\pm 300$	$\pm 585$	mV
$V_{SRON2}$	Receive Threshold Turn-On Voltage Reduced Threshold		$\pm 175$	$\pm 300$	mV
$V_{SROFF}$	Receive Threshold Turn-Off Voltage	(Note 1)	$\pm 175$	$\pm 300$	mV
$V_{DIFF}$	Differential Mode Input Voltage Range	$V_{CC} = 5.0$ V (Note 1)	-3.1	+3.1	V
$V_{OD}$	Differential Output Voltage	On-Chip Filters Selected 100 $\Omega$ Load	Typical: 5.0 (Notes 1, 2)		V pk-pk
<b>AUI</b>					
$V_{OD}$	Differential Output Voltage ( $TX \pm$ )	78 $\Omega$ Termination (Note 1)	$\pm 550$	$\pm 1200$	mV
$V_{OB}$	Differential Idle Output Voltage Imbalance ( $TX \pm$ )	78 $\Omega$ Termination (Note 1)	Typical: 40 mV		
$V_U$	Undershoot Voltage ( $TX \pm$ )	78 $\Omega$ Termination (Note 1)	Typical: 80 mV		
$V_{DS}$	Differential Squelch Threshold ( $RX \pm$ , $CD \pm$ )		-175	-300	mV
$V_{CM}$	Differential Input Common Mode Voltage ( $RX \pm$ , $CD \pm$ )	(Note 1)	0	5.25	V

**Note 1:** Not Tested in Production Test.

**Note 2:** In on-chip filter mode, the value of TPI  $V_{OD}$  when measured driving a 100 $\Omega$  load may not meet the IEEE 802.3 specification over temperature and process without the use of a resistor on RTX. This does not adversely effect system performance. For more details please refer to Applications Note AN-974.

# Preliminary Switching Design Guidelines

Memory Support Bus Accesses (for I/O port or FIFO transfers)



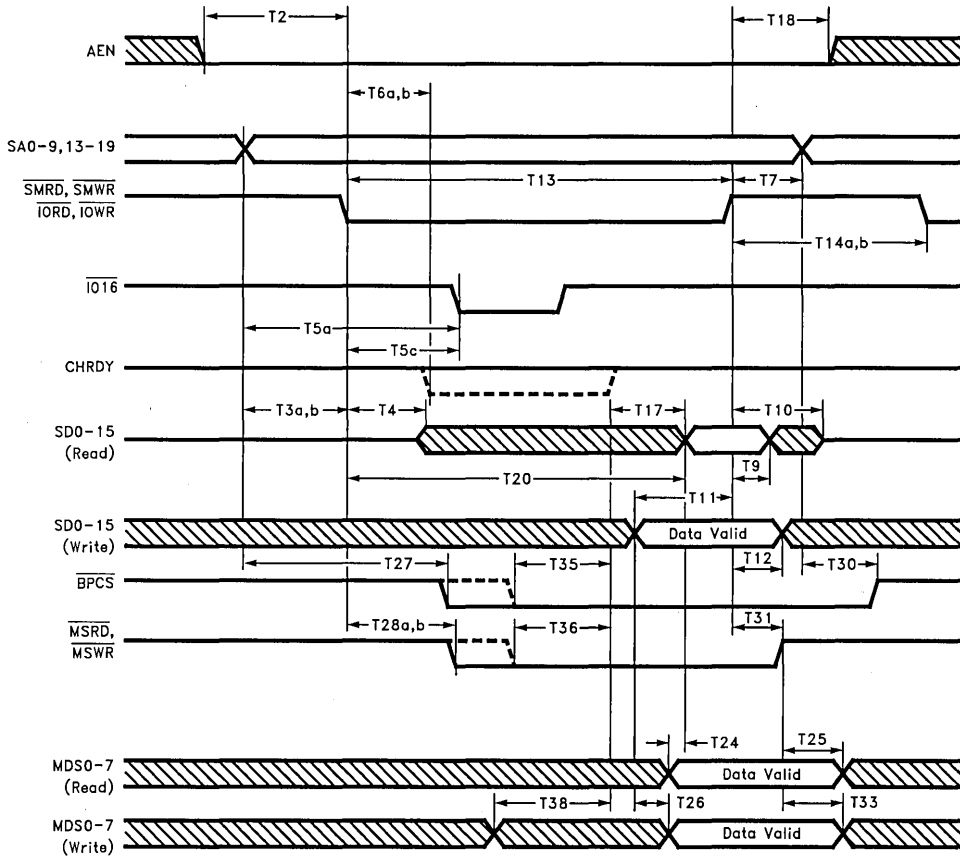
TL/F/12082-44

MEMORY SUPPLY BUS ACCESSES (for I/O Port or FIFO Transfers)

Symbol	Description	8-Bit Transfers		16-Bit Transfers		Units
		Min	Max	Min	Max	
T1	MSA1-13 Valid before RCS Asserted		30		30	ns
T2	MSA1-13 Valid before MSRDR-WR Asserted	20		20		ns
T3	MSRDR-WR Width	70		70		ns
T4	RCS Valid to MSWR or MSRDR Deasserted	105		105		ns
T5	MSA1-13 Valid after MSRDR-WR Deasserted	10		10		ns
T6	RCS Held after MSRDR-WR Deasserted	10		10		ns
T7	RCS Active to MSD(0:15) Valid		100		100	ns
T8	Read Data Hold from MSRDR Deasserted	0		0		ns
T9	Write Data Set-Up to MSWR Deasserted	40		40		ns
T10	Write Data Hold from MSWR Deasserted	10		10		ns

# Preliminary Switching Design Guidelines (Continued)

## ISA SLAVE ACCESSES



TL/F/12082-45

## Preliminary Switching Design Guidelines (Continued)

### ISA SLAVE ACCESSES (Continued)

Symbol	Description	8-Bit Transfers		16-Bit Transfers		Units
		Min	Max	Min	Max	
T2	AEN Valid before Command Strobe Active	60		60		ns
T3a	SA0–9 Valid before $\overline{\text{IORD}}$ , $\overline{\text{IOWR}}$ Asserted	40		20		ns
T3b	SA0–19 Valid before $\overline{\text{SMRD}}$ , $\overline{\text{SMWR}}$ Asserted	32				ns
T4	$\overline{\text{SMRD}}$ , $\overline{\text{IORD}}$ , Asserted to SD0–15 Driven (Notes 3, 14)	0		0		ns
T5a	SA0–9 Valid before $\overline{\text{IO16}}$ Valid (Notes 1, 9)				60	ns
T5c	SA0–9 Valid and $\overline{\text{IORD}}$ , $\overline{\text{IOWR}}$ Active before $\overline{\text{IO16}}$ Valid (Notes 1, 10)				50	ns
T6a	$\overline{\text{IORD}}$ , $\overline{\text{IOWR}}$ Asserted to CHRDY Negated (Notes 2, 5)		100		50	ns
T6b	$\overline{\text{SMRD}}$ , $\overline{\text{SMWR}}$ Asserted to CHRDY Negated (Note 2)		100			ns
T7	$\overline{\text{SMRD}}$ , $\overline{\text{SMWR}}$ , $\overline{\text{IORD}}$ , $\overline{\text{IOWR}}$ , Negated before SA0–9 Invalid	20		20		ns
T9	$\overline{\text{SMRD}}$ , $\overline{\text{IORD}}$ , Negated to SD0–15 Read Data Invalid (notes 3, 14)	0		0		ns
T10	$\overline{\text{SMRD}}$ , $\overline{\text{IORD}}$ negated to SD0–15 Floating (Note 3)		45		45	ns
T11	D0–15 Write Data Valid to $\overline{\text{IOWR}}$ , $\overline{\text{SMWR}}$ Negated (Note 3)	60			40	ns
T12	$\overline{\text{SMWR}}$ , $\overline{\text{IOWR}}$ , Negated to SD0–15 Write Data Invalid (Note 3)	20		20		ns
T13	$\overline{\text{SMRD}}$ , $\overline{\text{SMWR}}$ , $\overline{\text{IORD}}$ , $\overline{\text{IOWR}}$ Active Width (Note 8)	300		140		ns
T14a	$\overline{\text{IORD}}$ , $\overline{\text{IOWR}}$ Inactive Width	85		85		ns
T14b	$\overline{\text{SMRD}}$ , $\overline{\text{SMWR}}$ , Inactive Width	150		150		ns
T17	CHRDY Asserted to SD0–15 Read Data Valid (Notes 2, 3, 6)		60		60	ns
T18	$\overline{\text{SMRD}}$ , $\overline{\text{SMWR}}$ , $\overline{\text{IORD}}$ , $\overline{\text{IOWR}}$ Negated before AEN Invalid	25		25		ns
T20	$\overline{\text{SMRD}}$ , $\overline{\text{IORD}}$ Asserted to SD0–15 Read Data Valid (Notes 3 and 7)		150		90	ns
T24	Read Data Valid on MSD0–7 to Valid on SD0–7		70			ns
T25	$\overline{\text{MSRD}}$ Deasserted to MSD0–7 Read Data Invalid (Note 3)	0				ns
T26	Write Data Valid on SD0–7 to Valid on MSD0–7		90			ns



## Preliminary Switching Design Guidelines (Continued)

### ISA SLAVE ACCESSES (Continued)

Symbol	Description	8-Bit Transfers		16-Bit Transfers		Units
		Min	Max	Min	Max	
T27	SA0-19 Valid to BPCS Asserted (Note 11)		55	55		ns
T28a	MRD Asserted to MSR $\bar{D}$ Asserted		60	60		ns
T28b	MWR Asserted to MSWR $\bar{A}$ Asserted		120	120		ns
T30	SA0-19 Invalid to BPCS Negated (Note 11)	0				ns
T31	SMRD, Deasserted to MSR $\bar{D}$ Deasserted	0				ns
T33	MSWR Deasserted to MSD0-7 Invalid (Note 3)	0				ns
T35	BPCS Asserted to CHR $\bar{D}$ Y Asserted (Note 13)	175				ns
T36	MSRD Asserted to CHR $\bar{D}$ Y Asserted (Note 13)	150				ns
T38	MSD0-7 Asserted to CHR $\bar{D}$ Y Asserted (Note 13)	250				ns

**Note 1:** M $\bar{T}$ 6, I $\bar{O}$ T6 are only asserted for 16-bit transfers.

**Note 2:** CHR $\bar{D}$ Y is only deasserted if the NIC core can not service the access immediately. It is held deasserted until the NIC core is ready, causing the system to insert wait states.

**Note 3:** On 8-bit transfers only 8 bits of MSD0-15 and D0-7 are driven.

**Note 5:** This is the standard CHR $\bar{D}$ Y timing where CHR $\bar{D}$ Y is asserted after I $\bar{O}$ R $\bar{D}$  or I $\bar{O}$ W $\bar{R}$ .

**Note 6:** Read data valid is referenced to CHR $\bar{D}$ Y when wait states have been inserted.

**Note 7:** If no wait states are inserted read data valid can be measured from I $\bar{O}$ R $\bar{D}$ .

**Note 8:** This is a minimum timing with no additional wait states.

**Note 9:** This is the standard I $\bar{O}$ T6 timing where I $\bar{O}$ T6 is asserted after a valid address decode.

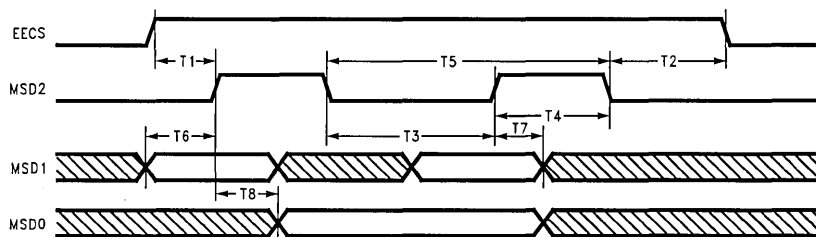
**Note 10:** This is the late I $\bar{O}$ T6 timing, required by some machines, where I $\bar{O}$ T6 is asserted after a valid address decode and I $\bar{O}$ R $\bar{D}$  or I $\bar{O}$ W $\bar{R}$  going active.

**Note 11:** This is a timing for a RAM access.

**Note 13:** This is a timing for a Boot PROM access.

**Note 14:** Guaranteed by design.

### SERIAL EEPROM TIMING



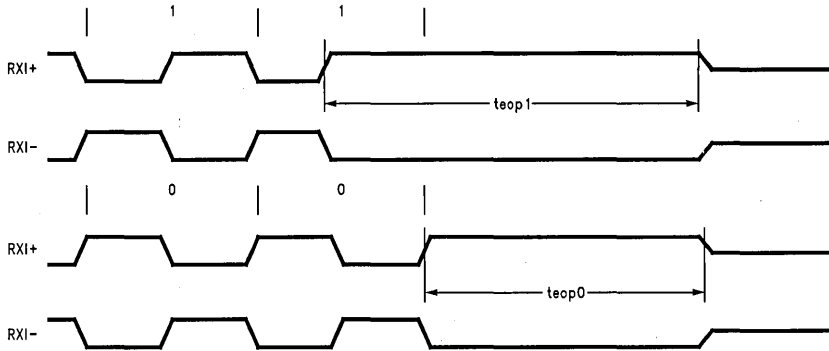
TL/F/12082-46

Symbol	Description	Min	Max	Units
T1	EECS Setup to MSD2	150		ns
T2	EECS Hold after MSD2	250		ns
T3	MSD2 Low Time	450		ns
T4	MSD2 High Time	450		ns
T5	MSD2 Clock Period (Note 1)	1		$\mu$ s
T6	MSD1 Setup to MSD2 High	100		ns
T7	MSD2 Hold from MSD2 High	100		ns
T8	MSD0 Valid from MSD2 High		500	ns

**Note 1:** Derived from crystal oscillator tolerance =  $\pm$  0.01%.

# Preliminary Switching Design Guidelines (Continued)

## TPI RECEIVE TIMING (End of Packet)

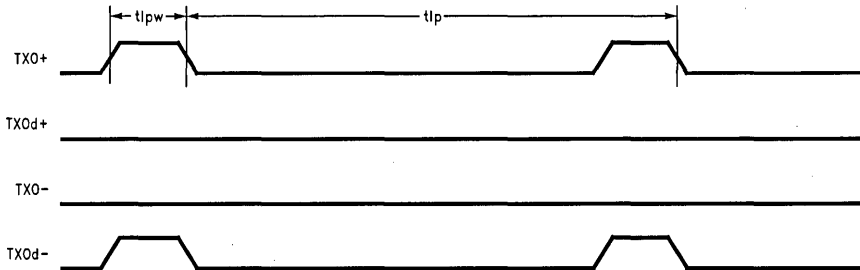


TL/F/12082-47

Symbol	Description	Min	Max	Units
$t_{eop1}$	Receive End of Packet Hold Time after Logic "1" (Note 1)	225		ns
$t_{eop0}$	Receive End of Packet Hold Time after Logic "0" (Note 1)	225		ns

**Note 1:** This parameter guaranteed by design and is not tested.

## LINK PULSE TIMING

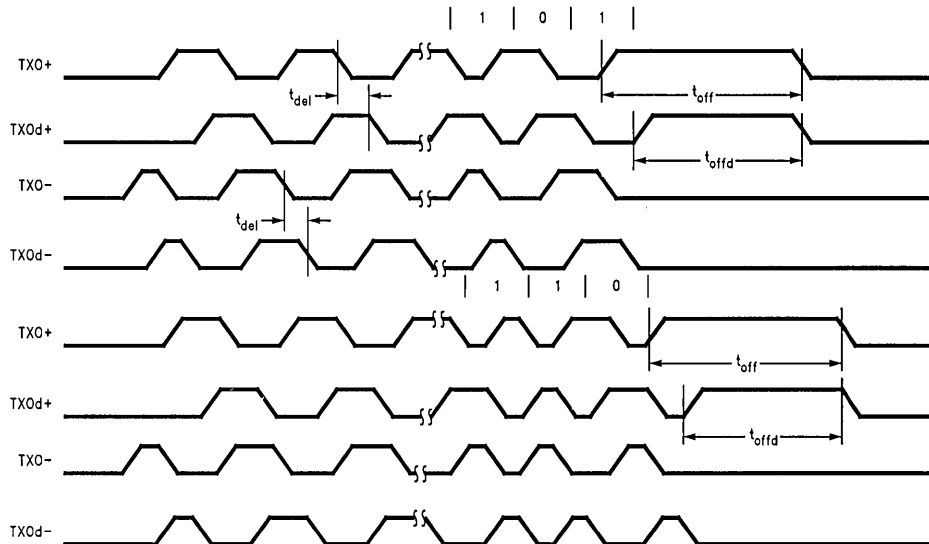


TL/F/12082-48

Symbol	Description	Min	Max	Units
$t_{lp}$	Time between Link Output Pulses	8	24	ns
$t_{lpw}$	Link Integrity Output Pulse Width	80	130	ns

# Preliminary Switching Design Guidelines (Continued)

## TPI TRANSMIT TIMING (End of Packet)



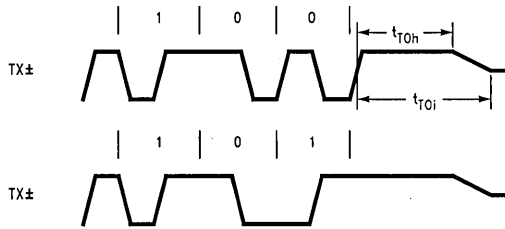
TL/F/12082-49

Symbol	Description	Min	Max	Units
$t_{del}$	Pre-Emphasis Output Delay (TXO± to TXOd±) (Note 1)	46	54	ns
$t_{Off}$	Transmit Hold Time at End of Packet (TXO±) (Note 1)	250		ns
$t_{Offd}$	Transmit Hold Time at End of Packet (TXOd±) (Note 1)	200		ns

**Note 1:** This parameter is guaranteed by design and is not tested.

# Preliminary Switching Design Guidelines (Continued)

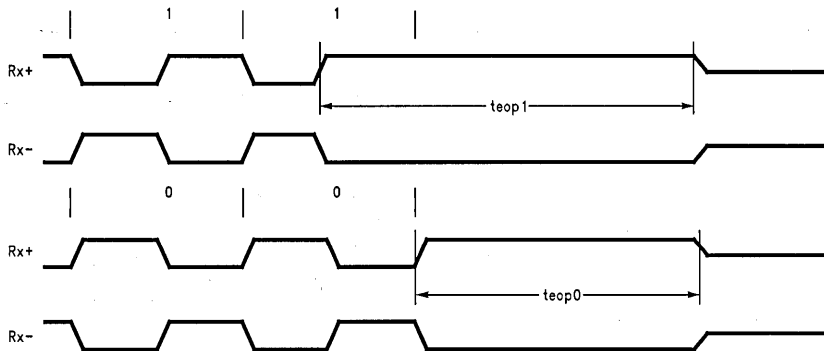
## AUI TRANSMIT TIMING (End of Packet)



TL/F/12082-50

Symbol	Description	Min	Max	Units
$t_{TOh}$	Transmit Output High before Idle (Note 1)	200		ns
$t_{TOl}$	Transmit Output Idle Time (Note 1)	8000		ns

## AUI RECEIVE TIMING (End of Packet)



TL/F/12082-51

Symbol	Description	Min	Max	Units
$t_{eop1}$	Receive End of Packet Hold Time after Logic "1" (Note 1)	225		ns
$t_{eop0}$	Receive End of Packet Hold Time after Logic "0" (Note 1)	225		ns

**Note 1:** This parameter is guaranteed by design and is not tested.

# DP83905 AT/LANTIC™

## AT Local Area Network Twisted-Pair Interface Controller

### General Description

The AT/LANTIC AT Local Area Network Twisted-pair Interface Controller is a CMOS VLSI device designed for easy implementation of CSMA/CD local area networks.

Unique to the AT/LANTIC is the integration of the entire bus interface for PC-AT® ISA (Industry Standard Architecture) bus based systems. Hardware and software selectable options allow the AT/LANTIC's bus interface to be configured software compatible to either an NE2000 or Ethercard PLUS16™. All bus drivers and control logic are integrated to reduce board cost and area.

Supported network interfaces include 10BASE5 or 10BASE2 Ethernet via an external transceiver connected to its AUI port, and Twisted-pair Ethernet (10BASE-T) using the on-board transceiver. The AT/LANTIC provides the Ethernet Media Access Control (MAC), Encode-Decode (ENDEC) with an AUI interface, and 10BASE-T transceiver functions in accordance with the IEEE 802.3 standards.

The AT/LANTIC's integrated 10BASE-T transceiver fully complies with the IEEE standard. This functional block incorporates the receiver, transmitter, collision, heartbeat, loop-back, jabber, and link integrity blocks as defined in the standard. The transceiver when combined with equalization resistors, transmit/receive filters, and pulse transformers provides a complete physical interface from the AT/LANTIC Controller's ENDEC module and the twisted pair medium.

(Continued)

### Features

- Controller and integrated bus interface solution for IEEE 802.3, 10BASE5, 10BASE2, and 10BASE-T
- Software compatible with *Novell®'s NE2000/Plus* industry standard Ethernet Adapters
- Selectable buffer memory size
- No external bus logic or drivers
- Integrated controller, ENDEC, and transceiver
- Full IEEE 802.3 AUI interface
- Single 5V supply

#### 10BASE-T TRANSCEIVER MODULE:

- Integrates transceiver functionality:
  - Transmitter and receiver functions
  - Collision detect, heartbeat and jabber
  - Selectable link integrity test or link disable
  - Polarity Detection/Correction

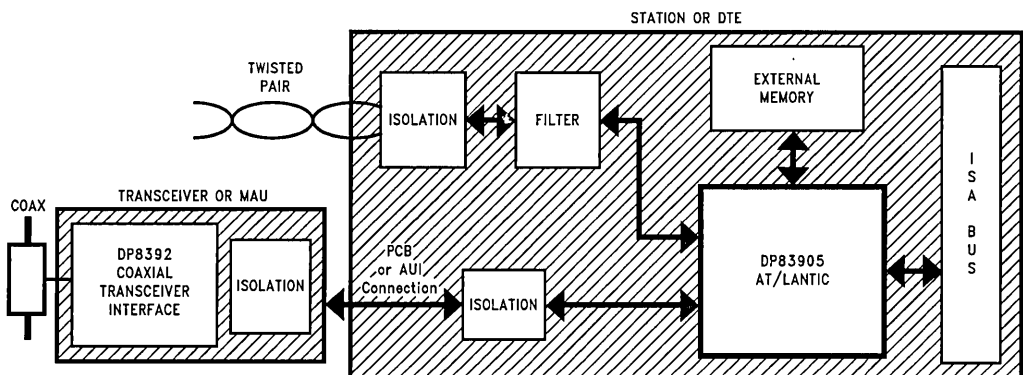
#### ENDEC MODULE:

- 10 Mbit/s Manchester encoding/decoding
- Squelch on receive and collision pairs

#### MAC/CONTROLLER MODULE:

- Software compatible with DP8390, DP83901, DP83902
- Efficient buffer management implementation

## 1.0 System Diagram



TL/F/11498-1

## General Description (Continued)

The integrated ENDEC module allows Manchester encoding and decoding via a differential transceiver and phase lock loop decoder at 10 Mbit/sec. Also included are a collision detect translator and diagnostic loopback capability. The ENDEC module interfaces directly to the transceiver module, and also provides a fully IEEE compliant AUI (Attachment Unit Interface) for connection to other media transceivers.

The Media Access Control function which is provided by the Network Interface Control module (NIC) provides simple and efficient packet transmission and reception control by means of off-board memory which can be accessed either through an I/O port or mapped into the system memory.

AT/LANTIC Controller provides a comprehensive solution for 10BASE-T IEEE 802.3 networks. Due to the inherent constraints of CMOS processing, isolation is required at the AUI differential signal interface for 10BASE5 and 10BASE2 applications.

## Table of Contents

### 1.0 SYSTEM DIAGRAM

- 1.1 Connection Diagram

### 2.0 PIN DESCRIPTION

### 3.0 SIMPLIFIED APPLICATION DIAGRAM

### 4.0 FUNCTIONAL DESCRIPTION

- 4.1 Bus Interface Block
- 4.2 Power on RESET operation
- 4.3 EEPROM Operation
- 4.4 Jumpered and Jumperless Operation Support
- 4.5 Low Power Operation
- 4.6 Boot PROM Operation
- 4.7 DP8390 Core (Network Interface Controller)
- 4.8 Twisted Pair Interface Module
- 4.9 Encoder/Decoder (ENDEC) Module

### 5.0 REGISTER DESCRIPTIONS

- 5.1 Configuration Registers
- 5.2 Shared Memory Mode Control Registers
- 5.3 NIC Core Registers
- 5.4 DP8390 Core DMA Registers

### 6.0 OPERATION OF AT/LANTIC CONTROLLER

- 6.1 Transmit/Receive Packet Encapsulation/Decapsulation
- 6.2 Buffer Memory Access Control (DMA)
- 6.3 Packet Reception
- 6.4 Packet Transmission
- 6.5 Loopback Diagnostics
- 6.6 Memory Arbitration and Bus Operation
- 6.7 Functional Bus Timing

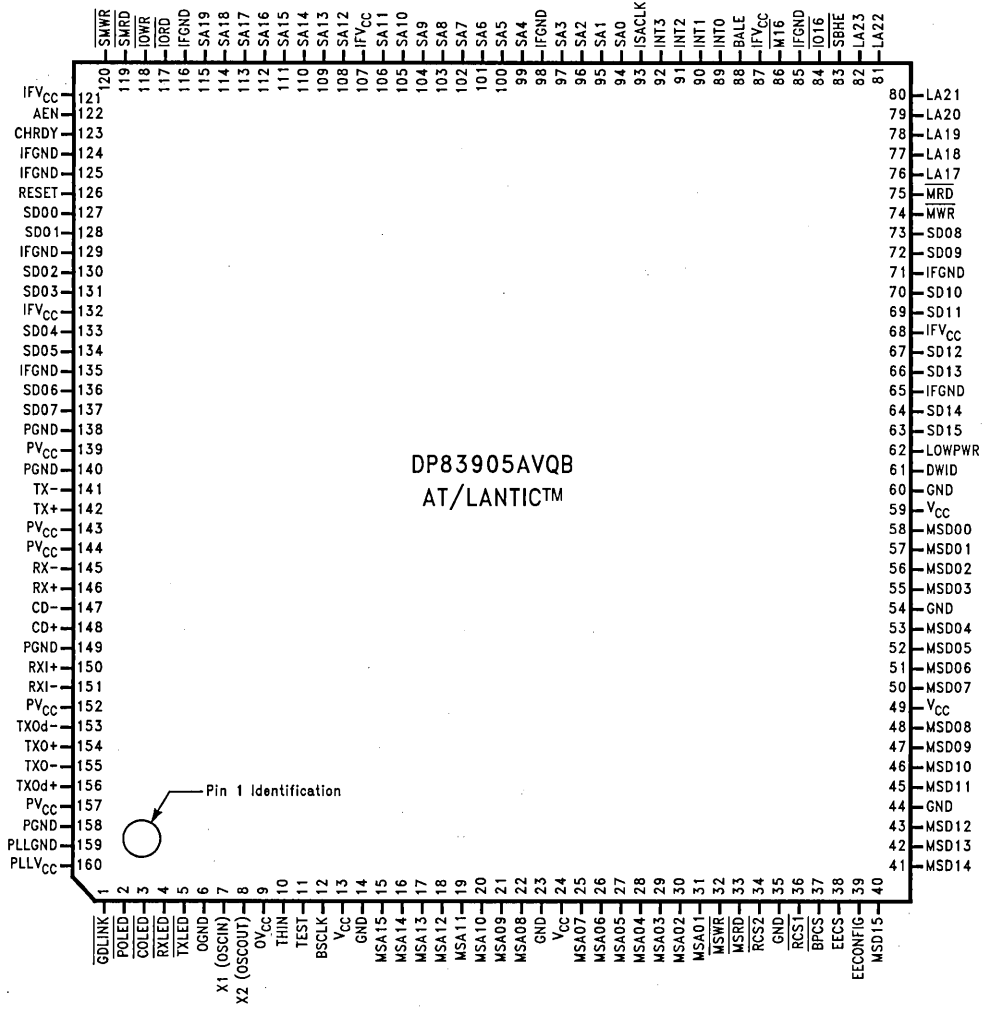
### 7.0 PRELIMINARY ELECTRICAL CHARACTERISTICS

### 8.0 PRELIMINARY SWITCHING CHARACTERISTICS

### 9.0 AC TIMING TEST CONDITIONS

# 1.0 System Diagram (Continued)

## 1.1 CONNECTION DIAGRAM



**Order Number DP83905AVQB  
See NS Package Number VUL160A**

TL/F/11498-2

## 2.0 Pin Description

Pin No.	Pin Name	Type*	Description
<b>ISA BUS INTERFACE PINS</b>			
94-97 99-106 108-115	SA0-SA19	I TTL	<b>LATCHED ADDRESS BUS:</b> Low-order 20 bits of the system's 24 bit address bus. These lines are enabled onto the bus, by the system, when BALE is high and are latched when BALE returns low. These bits are used to decode accesses to the AT/LANTIC Controller's I/O map and to the boot PROM. In addition they are used to decode accesses to the AT/LANTIC Controller's memory in shared memory mode.
76-82	LA17-LA23	I TTL	<b>UNLATCHED ADDRESS BUS:</b> High order 7 bits of the 24-bit system address bus. These lines are valid on the falling edge of BALE. These bits are used to decode accesses to the AT/LANTIC Controller's memory in shared memory mode.
127, 128, 130, 131, 133, 134, 136, 137, 73, 72, 70, 69, 67, 66, 64, 63.	SD0-SD15	I/O 3SH	<b>SYSTEM DATA BUS:</b> 16-bit system data bus. Used to transfer data between the system and the AT/LANTIC Controller.
88	BALE	I TTL	<b>BUS ADDRESS LATCH ENABLE:</b> This signal indicates when the system address lines are valid.
83	SBHE	I TTL	<b>SYSTEM BUS HIGH ENABLE:</b> This signal indicates that the system expects a transfer on the upper byte lane.
86	M16	O OCH	<b>16-BIT MEMORY TRANSFER:</b> In 16-bit shared memory mode this signal indicates that the AT/LANTIC Controller has decoded an address within the 128 kbyte space that it occupies part of.
84	IOT16	O OCH	<b>16-BIT I/O TRANSFER:</b> In I/O mode this signal indicates that the AT/LANTIC Controller is responding to a 16-bit I/O access by driving 16-bits of data on the bus.
74	MWR	I TTL	<b>MEMORY WRITE STROBE:</b> Strobe from system to write to AT/LANTIC Controller's memory map. <i>This pin should be connected to allow the CHRDY fix in 16-bit I/O mode to operate correctly. (See Section 6.0)</i>
75	MRD	I TTL	<b>MEMORY READ STROBE:</b> Strobe from system to read from AT/LANTIC Controller's memory map. <i>This pin should be connected to allow the CHRDY fix in 16-bit I/O mode to operate correctly. (See Section 6.0)</i>
119, 120	SMRD & SMWR	I TTL	<b>LOW MEMORY STROBES:</b> In Memory mode these signals strobe memory transfers in the same manner as MRD and MWR except that these signals only occur if the access is to the lowest 1 Megabyte. This partial address decode means that these signals can be used in an 8-bit slot to properly decode an access to this area. <i>The AT/LANTIC Controller will use MRD and MWR in 16-bit Memory mode and will use SMRD and SMWR in Memory mode when DWID is low (8-bit mode). SMRD and SMWR are also used to access the BOOT PROM.</i>
118	IOWR	I TTL	<b>I/O WRITE STROBE:</b> Strobe from system to write to the AT/LANTIC Controller's I/O map.
117	IORD	I TTL	<b>I/O READ STROBE:</b> Strobe from system to read from the AT/LANTIC Controller's I/O map.
126	RESET	I TTL	<b>RESET:</b> This signal is output by the system to reset all devices on the bus.

\*Driver Types are: I = Input, O = Output, I/O = Bi-directional Output, OCH = Open Collector, 3SH = TRI-STATE Output, TTL = TTL Compatible, AUI = Attachment Unit Interface, TPI = Twisted Pair Interface, LED = LED Drive, MOS = CMOS Level Compatible, XTAL = Crystal.



## 2.0 Pin Description (Continued)

Pin No.	Pin Name	Type*	Description
<b>ISA BUS INTERFACE PINS (Continued)</b>			
123	CHRDY	O OCH	<b>CHANNEL READY:</b> This signal is used to insert wait states into system accesses.
122	AEN	I TTL	<b>DMA ACTIVE:</b> This signal indicates that the system's DMA controller has control of the bus.
89–92	INT0–3	O 3SH	<b>INTERRUPT REQUEST:</b> The operation of these 4 outputs is determined by the Configuration registers. They can either be used to directly drive the interrupt lines or used as a 3-bit code with a strobe to generate up to 8 interrupts.
61	DWID	I MOS	<b>DATA WIDTH:</b> This input specifies whether the AT/LANTIC Controller is interfacing to an 8- or 16-bit ISA bus. When high it is in 16-bit mode. It has an internal pull-down resistor.
93	ISACK	I TTL	<b>ISA CLOCK:</b> Clock from ISA bus. This signal is only required if CHRDY timing has to be altered, by changing the CHRDY bit of Configuration Register B.
<b>NETWORK INTERFACE PINS</b>			
156–153	TX0d+, TX0–, TXO+, TXOd–	O TPI	<b>TWISTED PAIR TRANSMIT OUTPUTS:</b> These high drive CMOS level outputs are resistively combined external to the chip to produce a differential output signal with equalization to compensate for Intersymbol Interference (ISI) on the twisted pair medium.
150, 151	RX1+, RX1–	I TPI	<b>TWISTED PAIR RECEIVE INPUTS:</b> These inputs feed a differential amplifier which passes valid data to the ENDEC module.
141, 142	TX– TX+	O AUI	<b>AUI TRANSMIT OUTPUT:</b> Differential driver which sends the encoded data to the transceiver. The outputs are source followers which require 270Ω pull-down resistors.
145, 146	RX– RX+	I AUI	<b>AUI RECEIVE INPUT:</b> Differential receive input pair from the transceiver.
147, 148	CD– CD+	I AUI	<b>AUI COLLISION INPUT:</b> Differential collision pair input from the transceiver.
5	$\overline{\text{TXLED}}$	O LED	<b>TRANSMIT:</b> An open-drain active low output. It is asserted for approximately 50 ms whenever the AT/LANTIC Controller transmits data in either AUI or TPI modes.
4	$\overline{\text{RXLED}}$	O LED	<b>RECEIVE:</b> An open-drain active low output. It is asserted for approximately 50 ms whenever receive data is detected in either AUI or TPI mode.
3	$\overline{\text{COLED}}$	O LED	<b>COLLISION:</b> An open-drain active low output. It is asserted for approximately 50 ms whenever the AT/LANTIC Controller detects a collision in either AUI or TPI modes.
1	$\overline{\text{GDLNK}}$	O LED	<b>GOOD LINK:</b> An open-drain active low output. This pin operates as an output to display link integrity status if this function has not been disabled by the GDLNK bit in Configuration Register B. This output is off if the AT/LANTIC Controller is in AUI mode or if link testing is enabled and the link integrity is bad (i.e. the twisted pair link has been broken). This output is on if the AT/LANTIC Controller is in Twisted Pair Interface (TPI) mode, link integrity checking is enabled and the link integrity is good (i.e. the twisted pair link has <b>not</b> been broken) or if the link testing is disabled.
2	$\overline{\text{POLED}}$	O LED	<b>POLARITY:</b> An open-drain active low output. This signal is normally inactive. When the TPI module detects seven consecutive link pulses or three consecutive received packets with reversed polarity $\overline{\text{POLED}}$ is asserted.

\*Driver Types are: I = Input, O = Output, I/O = Bi-directional Output, OCH = Open Collector, 3SH = TRI-STATE Output, TTL = TTL Compatible, AUI = Attachment Unit Interface, TPI = Twisted Pair Interface, LED = LED Drive, MOS = CMOS Level Compatible, XTAL = Crystal.

## 2.0 Pin Description (Continued)

Pin No.	Pin Name	Type*	Description
<b>NETWORK INTERFACE PINS (Continued)</b>			
7	X1 (OSCIN)	I XTAL	<b>CRYSTAL OR EXTERNAL OSCILLATOR INPUT</b>
8	X2 (OSCOU)	O XTAL	<b>CRYSTAL FEEDBACK OUTPUT:</b> Used in crystal connections only. Should be left completely unconnected when using an oscillator module.
10	THIN	O DCDC	<b>THIN CABLE:</b> This output is high if AT/LANTIC Controller is configured for thin cable. It can be used to enable the DC-DC converter required by the thin ethernet configuration.
<b>EXTERNAL MEMORY SUPPORT</b>			
58-50	MSD0-7, CA0-7, DO, DI, SK	I/O, I, O MOS	<b>MEMORY SUPPORT DATA BUS—CONFIGURATION REGISTER A INPUT EEPROM SIGNALS:</b> MSD0-7: When RESET is inactive these pins can be used to access external memory and boot PROM. CA0-7: When RESET is active Configuration Register A is loaded with the data value on these pins. If the user puts an external pull-up on any of these pins then the corresponding register bit is set to a 1. If the pin is left unconnected then the register bit is 0. DO, DI, SK: When RESET goes from an active to an inactive level AT/LANTIC Controller will read the contents of an EEPROM, using these signals, and load the contents into internal registers. These internal registers will then be mapped into the space taken up by the PROM in the NE2000 and Ethercard PLUS16. After the EEPROM read operation has completed these pins will revert to MSD0-2 (D0 = MSD0, DI = MSD1, SK = MSD2).
48-45 43-40	MSD8-15 or CB0-7	I/O, I MOS	<b>MEMORY SUPPORT DATA BUS—CONFIGURATION REGISTER B INPUT:</b> MSD8-15: When RESET is inactive these pins can be used to access external memory. CB0-7: When RESET is active Configuration Register B is loaded with the data value on these pins. If the user puts an external pull-up on any of these pins then the corresponding register bit is set to a 1. If the pin is left unconnected then the register bit is 0.
31-25, 22	MSA1-8 or CC0-7	O, I MOS	<b>MEMORY SUPPORT ADDRESS BUS—CONFIGURATION REGISTER C INPUT:</b> MSA1-8: When RESET is inactive these pins drive the memory support address bus. CC0-7: When RESET is active Configuration Register C is loaded with the data value on these pins. If the user puts an external pull-up on any of these pins then the corresponding register bit is set to a 1. If the pin is left unconnected then the register bit is 0.
21-15	MSA9-15	O MOS	<b>MEMORY SUPPORT ADDRESS BUS:</b> MSA9-15: When RESET is inactive these pins drive the memory support address bus. When the memory is only 8 bits wide A0 will appear on A13, in compatible mode, and on A15, in non-compatible mode.
33	MSRD	O MOS	<b>MEMORY SUPPORT BUS READ:</b> Strokes data from the external RAM into the AT/LANTIC Controller via the memory support data bus.
32	MSWR	O MOS	<b>MEMORY SUPPORT BUS WRITE:</b> Strokes data from the AT/LANTIC Controller into the external RAM via the memory support data bus.
37	BPCS	O MOS	<b>BOOT PROM CHIP SELECT:</b> Selects the boot PROM on the memory support data bus.

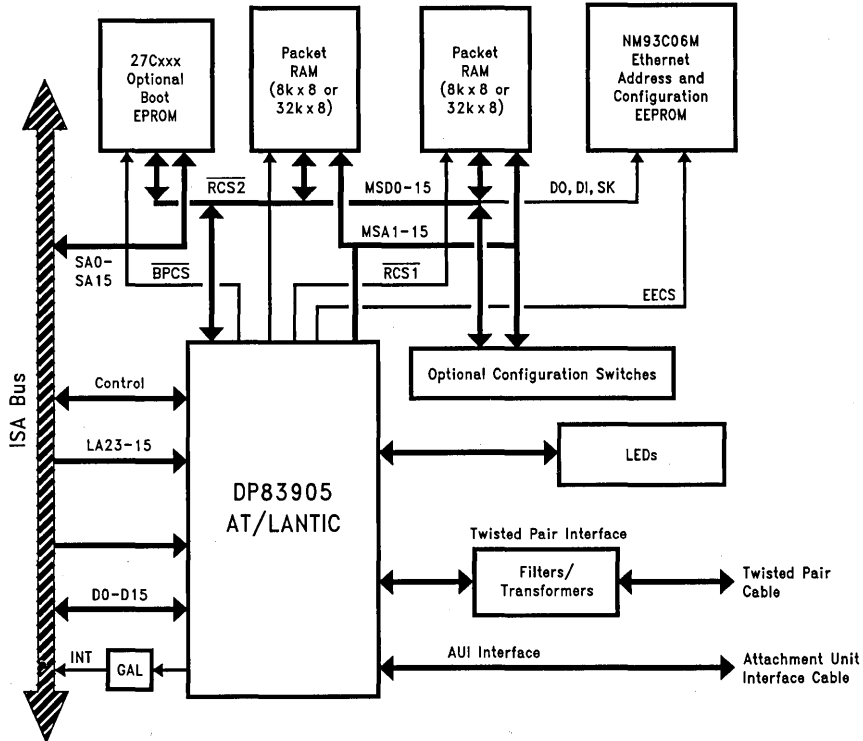
\*Driver Types are: I = Input, O = Output, I/O = Bi-directional Output, OCH = Open Collector, 3SH = TRI-STATE Output, TTL = TTL Compatible, AUI = Attachment Unit Interface, TPI = Twisted Pair Interface, LED = LED Drive, MOS = CMOS Level Compatible, XTAL = Crystal.

## 2.0 Pin Description (Continued)

Pin No.	Pin Name	Type*	Description
<b>EXTERNAL MEMORY SUPPORT (Continued)</b>			
33	$\overline{RCST}$	O MOS	<b>RAM CHIP SELECT 1:</b> Drives the chip select of the external RAM on the lower half of the memory support data bus.
34	$\overline{RCS2}$	O MOS	<b>RAM CHIP SELECT 2:</b> Drives the chip select of the external RAM on the upper half of the memory support data bus.
38	EECS	O MOS	<b>EEPROM CHIP SELECT:</b> Strobes data from the EEPROM onto the memory support data bus.
39	EECONFIG	I TTL	<b>CONFIGURE FROM EEPROM:</b> When this pin is tied high the AT/LANTIC Controller loads the configuration from an EEPROM.
12	BSCLK	I TTL	<b>INTERNAL BUS CLOCK:</b> This controls the speed of the NIC core if it is not running off of an internal clock (see Configuration Register C). This pin should be tied to ground if it is unused.
<b>LOW POWER SUPPORT</b>			
62	LOWPWR	I TTL	<b>LOW POWER:</b> Instructs AT/LANTIC Controller to enter its low power mode, as detailed in Section 4.5. Should be tied to ground for normal operation.
<b>TEST SUPPORT</b>			
11	TEST	I MOS	<b>TEST:</b> This input is only used for test mode. It should be left unconnected as it has an internal pull-down resistor which will enable correct operation.
<b>POWER SUPPLY PINS</b>			
160	PLL <sub>VCC</sub>		<b>PLL 5V SUPPLY PINS:</b> This pin supplies 5V to the AT/LANTIC's analog PLL inside the ENDEC block. To maximize data recovery it is recommended that analog layout and decoupling rules be applied between this pin and PLL <sub>GND</sub> .
159	PLL <sub>GND</sub>		<b>PLL NEGATIVE (GROUND) SUPPLY PINS</b>
157, 152, 144, 143, 139	PV <sub>CC</sub>		<b>PHYSICAL MEDIA 5V SUPPLY PINS:</b> These pins supply 5V to the AT/LANTIC's analog physical media interface circuitry.
158, 149, 140, 138	PG <sub>ND</sub>		<b>PHYSICAL LAYER NEGATIVE (GROUND) SUPPLY PINS:</b> These pins are the ground to the AT/LANTIC's analog physical media interface circuitry.
9	OV <sub>CC</sub>		<b>OSCILLATOR 5V SUPPLY PINS:</b> This pin supplies 5V to the AT/LANTIC's oscillator and LED circuitry.
6	OG <sub>ND</sub>		<b>OSCILLATOR NEGATIVE (GROUND) SUPPLY PINS:</b> This pin is the ground to the AT/LANTIC's oscillator and LED circuitry.
59, 49, 24, 13	V <sub>CC</sub>		<b>POSITIVE 5V SUPPLY PINS:</b> These pins supply power to the AT/LANTIC Controller's logic.
60, 54, 44, 35, 23, 14	G <sub>ND</sub>		<b>NEGATIVE (GROUND) SUPPLY PINS:</b> These are the supply pins for the AT/LANTIC Controller's logic. It is suggested that decoupling capacitors be connected between the V <sub>CC</sub> and G <sub>ND</sub> pins. It is essential to provide a path to ground for the G <sub>ND</sub> pins with the lowest possible impedance.
132, 121, 107, 87, 68	IFV <sub>CC</sub>		<b>INTERFACE POSITIVE 5V SUPPLY PINS:</b> These pins supply power to the AT/LANTIC Controller's ISA interface.
135, 129, 125, 124, 116, 98, 85, 71, 65	IFG <sub>ND</sub>		<b>INTERFACE NEGATIVE (GROUND) SUPPLY PINS:</b> These are the supply pins for the AT/LANTIC Controller's ISA interface. It is suggested that decoupling capacitors be connected between the IFV <sub>CC</sub> and IFG <sub>ND</sub> pins. It is essential to provide a path to ground for the IFG <sub>ND</sub> pins with the lowest possible impedance.

\*Driver Types are: I = Input, O = Output, I/O = Bi-directional Output, OCH = Open Collector, 3SH = TRI-STATE Output, TTL = TTL Compatible, AUI = Attachment Unit Interface, TPI = Twisted Pair Interface, LED = LED Drive, MOS = CMOS Level Compatible, XTAL = Crystal.

### 3.0 Simplified Application Diagram



TL/F/11498-3

### 4.0 Functional Description

The AT/LANTIC Controller is a highly integrated and configurable Ethernet controller making it suitable for most Ethernet applications. The AT/LANTIC Controller integrates the functions of the following blocks:

1. DP8390 Ethernet Controller Core and Media Access Control logic.
2. ISA Bus Interface containing all logic required to connect the DP8390 core to a packet buffer RAM and the ISA bus.
3. Media Interface which includes an Encoder/Decoder block with an AUI (Attachment Unit Interface) and a 10BASE-T Twisted Pair Interface.

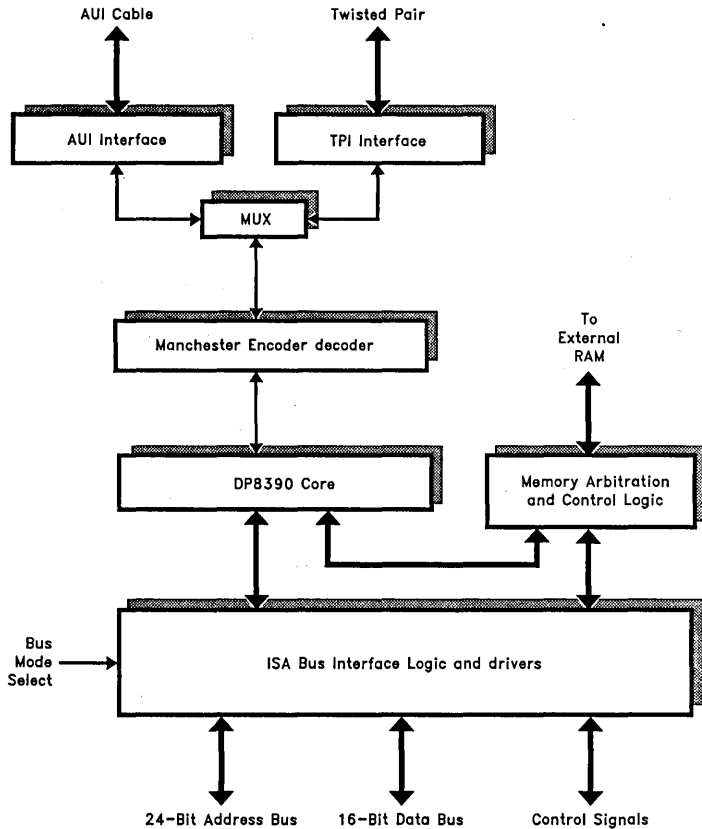
#### 4.1 BUS INTERFACE BLOCK

The AT/LANTIC Controller's Bus interface block provides the circuitry to interface the Ethernet controller logic, and the external packet buffer RAM to an ISA (Industry Standard Architecture) Bus. The bus interface provides several configuration modes which offer various different features depending on the designer's specific design requirements. The possible modes are:

1. 16-Bit or 8-Bit Shared Memory Compatible Mode
2. 16-Bit or 8-Bit Shared Memory Enhanced Mode
3. 16-Bit or 8-Bit I/O Port Compatible Mode
4. 16-Bit or 8-Bit I/O Port Enhanced Mode

This section describes the function of each of these modes.

### 4.0 Functional Description (Continued)



**FIGURE 1. Block Diagram of AT/LANTIC Controller**

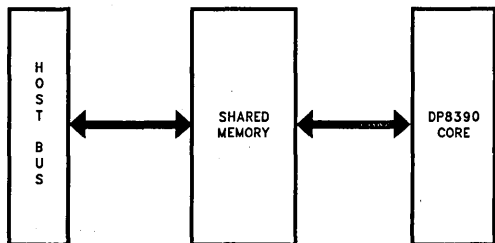
TL/F/11498-4

#### DETERMINING 8- OR 16-BIT WIDE DATA

AT/LANTIC Controller can treat the system data bus and all internal data busses as 8 or 16 bits wide. 8- or 16-bit mode is determined by the DWID pin. For an adapter card this bit can be used to automatically detect if the card has been plugged into an 8- or 16-bit slot. If this pin is connected to a V<sub>DD</sub> on the upper connector it will be high when plugged into a 16-bit slot, enabling 16-bit mode, and floating when plugged into an 8-bit slot. When floating the internal pull-down resistor will enable 8-bit mode.

In this mode the AT/LANTIC Controller's internal memory map, using external RAM devices, is mapped into the host system's memory map. Both the AT/LANTIC Controller and the host system can directly access this memory. The AT/LANTIC Controller controls the arbitration for this memory area, giving priority to its internal accesses. It also has an internal FIFO to allow for any latency on internal transfers introduced by system accesses. If a system access occurs while an internal access is current the AT/LANTIC Controller will insert wait states into the system cycle until the internal transfer is complete.

#### SHARED MEMORY ARCHITECTURE



TL/F/11498-5

**FIGURE 2. Shared Memory**

In this mode the AT/LANTIC Controller's internal registers are accessed within the system's I/O map. The address within this I/O map is set by Configuration Register A. The user programs the address of the shared memory within the host systems memory map by writing to a register in AT/LANTIC Controller. The memory is not accessible by the user until after this register has been programmed.

There are two basic Shared Memory modes, compatible mode, and non-compatible mode, as described in the following text.

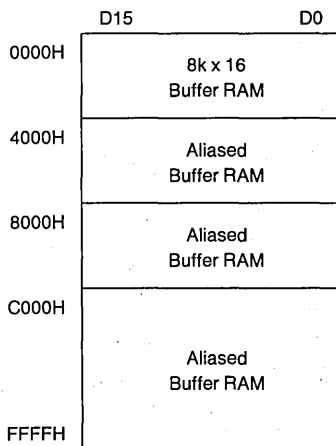
## 4.0 Functional Description (Continued)

### Shared Memory Compatible Mode I/O Address Mapping

The shared memory is at an address decided by the Address Decode Register and the base I/O address of AT/LANTIC Controller is configured in Configuration Register A. At that address the following structure appears.

Addr	D7-0	
00	Control 1	
01	AT detect	(Read only)
02	Unused	
03	Unused	
04	Unused	
05	Control 2	
06	Unused	
07	Unused	
08	Node addr 0	(Read only)
09	Node addr 1	(Read only)
0A	Node addr 2	(Read only)
0B	Node addr 3	(Read only)
0C	Node addr 4	(Read only)
0D	Node addr 5	(Read only)
0E	05h	(Read only)
0F	Checksum	(Read only)
10 to 1F	NIC registers	

(a)



(b)

**FIGURE 3. Shared Memory Mode a) Register Mapping and b) NIC Core Memory Map**

The AT Detect Register indicates whether AT/LANTIC Controller is in an 8- or 16-bit slot. The least significant bit of this register is set high when AT/LANTIC Controller is in 16-bit mode and low in 8-bit mode. Addresses 08H to 10H are

specified as the PROM space for compatibility with the Ethercard PLUS16. This is actually an array of 8-bit registers which are loaded from an external EEPROM after AT/LANTIC Controller is initialized by a reset pulse. The user should program the EEPROM to contain these values. The 8k words of memory can be accessed directly by the host system in the same manner as any other memory. Typically the programmer would remove data from this buffer using a "MOV" or "MOVSW" instruction.

### 8-BIT SHARED MEMORY COMPATIBLE MODE

In this mode the I/O map remains the same. The NIC core can still operate in 16-bit mode, if bit 6 of Control Register 2 is set high and the full 16 kbytes of RAM are still available. However, only 8-bit system accesses are allowed. If bit 6 of Control register 2 is low the NIC core must operate in 8-bit mode and only 8k of memory is available. The NIC Core data width is set by the WTS bit in the Data Configuration Register.

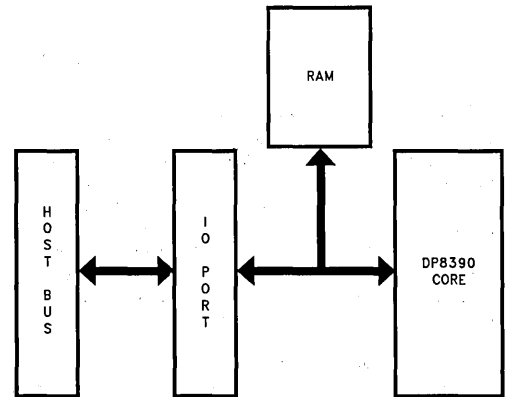
A low cost card, using only one 8 kbyte RAM, can be designed. If the DWID pin is left unconnected, or tied to GND, then the AT/LANTIC Controller will always operate in 8-bit mode, regardless of the slot the board is in.

If DWID is low the address bits of Control Register 2 should not be written to as they have no effect. In this mode the address comparator assumes that SA19 is to be compared to a logic high, with the other address comparisons programmed into Control Register 1.

### SHARED MEMORY NON-COMPATIBLE MODE

These modes are similar to the compatible mode. The difference is that they map a full 64 kbytes of RAM into the PC's memory address space. The I/O map remains the same.

### I/O PORT ARCHITECTURE



TL/F/11498-6

**FIGURE 4. I/O Port**

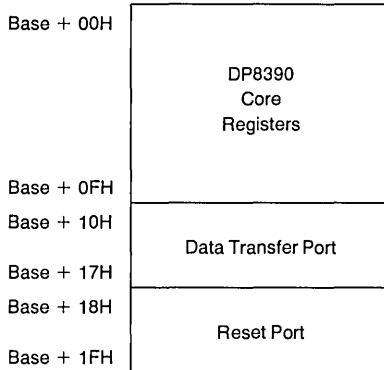
This is the architecture used by Novell's NE2000. In this mode the AT/LANTIC Controller's internal memory map is accessed byte or word at a time, via a port within the system's I/O space. AT/LANTIC Controller is programmed by the user to control the transfers between its internal memory and the I/O port.

In this mode the AT/LANTIC Controller's internal registers and the memory access port are accessed within the system's I/O map. The address within this I/O map is set by Configuration Register A.

## 4.0 Functional Description (Continued)

### 16-BIT I/O PORT COMPATIBLE MODE I/O ADDRESS MAPPING

This mode is compatible with Novell's NE2000. The base I/O address of the AT/LANTIC Controller is configured by Configuration Register A (either upon power up or by software writing to this register). At that address the following structure appears.

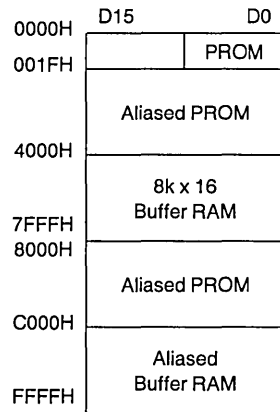


**FIGURE 5. I/O Port Mode Register I/O Map**

The registers within this area are 8 bits wide, but the data transfer port is 16 bits wide. The AT/LANTIC Controller's registers can be programmed to control the passing of data between its internal memory and the data transfer port. By accessing the data transfer port (using I/O instructions) the user can transfer data to or from the AT/LANTIC Controller's internal memory. The AT/LANTIC Controller's internal memory map is as shown in *Figure 6*.

AT/LANTIC Controller actually has a 64k address range but only does partial decoding on these devices. The PROM data is mirrored at all decodes up to 4000H and the entire map is repeated at 8000H. To access either the PROM or the RAM the user must initiate a Remote DMA transfer between the I/O port and memory.

On a remote read the AT/LANTIC Controller moves data from its internal memory map to the I/O port and the host system reads it by using an "INW" or "INSW" instruction from the I/O address of the data transfer port. If the system attempts to read the port before AT/LANTIC Controller has written the next word of data to it AT/LANTIC Controller will insert wait states into the system cycle, using the CHRDY



**(a)**

	D15	D0
1EH	00	57H
1CH	00	57H
	• 00 •	• RESERVED •
0AH	00	E'net Address 5
08H	00	E'net Address 4
06H	00	E'net Address 3
04H	00	E'net Address 2
02H	00	E'net Address 1
00H	00	E'net Address 0

**(b)**

**FIGURE 6. a) NIC Core's Memory Map  
b) 16-Bit Prom Map**

## 4.0 Functional Description (Continued)

line. AT/LANTIC Controller will not begin the next memory read until the previous word of data has been read.

On a remote write the system writes data to the I/O port, using an "OUTW" or "OUTSW" instruction, and AT/LANTIC Controller moves it to its buffer memory. If the system attempts to write to the port before AT/LANTIC Controller has moved the data to memory AT/LANTIC Controller will insert wait states into the system cycle, using the CHRDY line. AT/LANTIC Controller will not begin the next memory write until a new word has been written to the I/O port.

Addresses 00H to 1FH are specified as the PROM space for compatibility with the NE2000. This is actually an array of 8-bit registers which are loaded from an external EEPROM after AT/LANTIC Controller is initialized by an ISA RESET. They should contain the same data as the PROM did in the NE2000 and in the same format. As can be seen the PROM registers are only 8-bits wide. To transfer the data out the user must initiate a 16-bit DMA read transfer and discard the most significant byte of data on each transfer.

At address 00H of the PROM is a six byte Ethernet address for this node. The upper two addresses of the PROM store contain bytes which identify whether the AT/LANTIC Controller is in 8- or 16-bit mode. For 16-bit mode these bytes both contain the value 57H, for 8-bit mode they both contain 42H.

### 8-BIT I/O PORT COMPATIBLE MODE

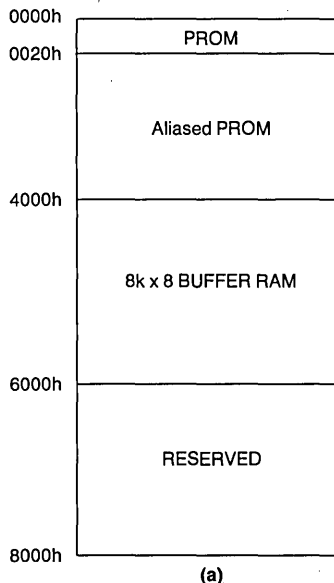
This mode is compatible with the 8-bit mode offered by Novell's NE2000. The NE2000 automatically detects whether it is in an 8- or 16-bit slot and configures itself appropriately. As explained in the previous paragraphs, the user can determine whether the board is in 8- or 16-bit mode by reading the PROM. In 8-bit mode only 8 kbytes of RAM are addressable, as in the 8-bit mode of the NE2000. The I/O map is the same as the 16-bit mode, the memory map is shown in Figure 7. Again the PROM has only a partial decode, so is mirrored at all addresses up to 4000H. The PROM still occupies 32 bytes of address space, although it only has 16 bytes of data, as the data at all odd address locations is merely a mirror of the data at the previous even address location. The RAM is mirrored at 6000H and the entire map mirrored at 8000H.

A low cost card, using only one 8 kbyte RAM, can be designed. If the DWID pin is left unconnected, or tied to GND, then the AT/LANTIC Controller will always operate in 8-bit mode, regardless of the slot the board is in.

### I/O PORT NON-COMPATIBLE MODE

This mode is similar to Novell's NE2000, but this mode allows the user to use the full 64 kbytes of address space except for an initial page for the PROM. The memory map for this board is shown in Figure 8. The memory map is the same for both 8- and 16-bit modes. Although the PROM store occupies 256 bytes, it is only 16 bytes long. The entire map is mirrored at 8000H.

A low cost card, using only one 8 kbyte RAM, can be designed. If the DWID pin is left unconnected, or tied to GND, then the AT/LANTIC Controller will always operate in 8-bit mode, regardless of the slot the board is in.



(a)

	D15	D0
1EH	42H	42H
1CH	42H	42H
	• RESERVED •	• RESERVED •
0AH	E'net Address 5	E'net Address 5
08H	E'net Address 4	E'net Address 4
06H	E'net Address 3	E'net Address 3
04H	E'net Address 2	E'net Address 2
02H	E'net Address 1	E'net Address 1
00H	E'net Address 0	E'net Address 0

(b)

FIGURE 7. a) 8-Bit NIC Core's Memory Map  
b) 8-Bit PROM Map

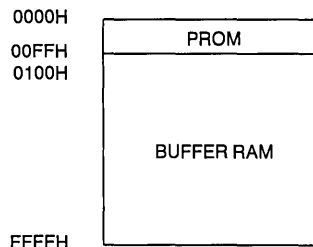


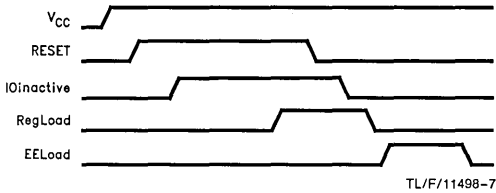
FIGURE 8. I/O Port Enhanced Mode  
DP8390 Core Memory Map



## 4.0 Functional Description (Continued)

### 4.2 POWER ON RESET OPERATION

The AT/LANTIC Controller configures itself after a Reset signal is applied. To be recognized as a valid Power-On-Reset the Reset signal must be active for at least 415  $\mu$ s. Figure 9 shows how the RESET circuitry operates.



**FIGURE 9. RESET Operation**

The ISA standard determines that within 500 ns of RESET going active all devices should enter the appropriate reset condition. The AT/LANTIC Controller will generate the internal signal IOinactive after RESET has been active for 415 ns, which will disable all outputs and cause RESET to be the only input monitored. The AT/LANTIC Controller will not respond to a RESET pulse of shorter duration than this. An internal timer continues to monitor the amount of time RESET is active. After 415  $\mu$ s it is considered a valid Power-On-Reset and an internal signal called RegLoad is generated.

When a Power-On-Reset occurs the AT/LANTIC Controller latches in the values on the configuration pins and uses these to configure the internal registers and options. Internally these pins contain pull-down resistors, which are enabled when IOinactive goes active. If any pins are unconnected they default to a logic zero. The internal pull-down resistor has a high resistance to allow the external pull-up resistors to be of a high value. This limits the current taken by the memory support bus. The suggested external resistor value is 10 k $\Omega$ . The configuration registers are loaded from the memory support bus when RESET goes inactive if RegLoad is active. The internal pull-down resistors are enabled onto the bus until RegLoad has gone inactive.

A Power-On-Reset also causes the AT/LANTIC Controller to load the internal PROM store from the EEPROM, which can take up to 320  $\mu$ s. This occurs after RegLoad has gone inactive. The AT/LANTIC Controller will be inaccessible during this time. If EECONFIG is held high the configuration data loaded on the falling edge of RESET will be overwritten with data read from the serial EEPROM. Regardless of the level on EECONFIG the PROM store will always be loaded with data from the serial EEPROM during the time specified as EELoad.

### 4.3 EEPROM OPERATION

The AT/LANTIC Controller uses an NM93C06, or EEPROM with compatible timings. The NM93C06 is a 256-bit device, arranged as 16 words each 16 bits wide. The programmed contents of the EEPROM is shown in Figure 10.

#### Mapping EEPROM Into PROM Space

Data is read from the EEPROM at boot time and stored in registers within the AT/LANTIC Controller. While this operation takes place the AT/LANTIC Controller can not be ac-

	D15	D0
0FH	73H	Config. C
0EH	Config B	Config. A
	•	•
	•	•
08H	42H	42H
07H	57H	57H
	•	•
	•	•
03H	Reserved (Checksum)	Reserved (Board Type)
02H	E'net Address 5	E'net Address 4
01H	E'net Address 3	E'net Address 2
00H	E'net Address 1	E'net Address 0

**Note 1:** The contents of locations 03H and 04H differ between I/O Mode and Shared Memory Mode. The Shared Memory Mode values are shown in parentheses. For compatibility with both modes default to the shared memory mode values.

**Note 2:** Programming 73H into the upper address is not absolutely required but is strongly recommended for future compatibility of manufacturing process.

**FIGURE 10. EEPROM Programming Map**

cessed by the system. These registers are mapped into the space traditionally occupied by the PROM in the NE2000 or the EtherCard PLUS16. The size and format of this data read is determined by the mode of operation.

#### SHARED MEMORY MODE

In this mode, program the EEPROM to contain the node's Ethernet address in the first six bytes, a byte identifying the type of board AT/LANTIC Controller is emulating in byte 7 and a checksum byte in byte 8. The two's complement sum of these eight bytes should equal FFH.

In this Mode the AT/LANTIC Controller reads the first 4 words from the EEPROM and maps them into the I/O map at the appropriate address.

#### I/O PORT MODE

In this mode, program the EEPROM to contain the node's Ethernet address in the first six bytes. The user should then program 5757H and 4242H into the subsequent bytes. The AT/LANTIC Controller will decide which of these values should be loaded into the PROM store depending on the DWID pin. (The data width is programmed in this mode by setting the WTS bit in the Data Configuration Register and setting the DWID pin for the proper mode.) If some other numerical values are preferred to indicate the mode then they can be programmed at this location in the EEPROM and AT/LANTIC Controller will put them at the correct address.

In this mode the AT/LANTIC Controller reads the first 7 words from the EEPROM and maps them into the memory map at the appropriate address. If in 16-bit mode it also

## 4.0 Functional Description (Continued)

reads the next word in the EEPROM and appends this. If in 8-bit mode it skips a word, then reads and appends the next word.

### Storing and Loading Configuration from EEPROM

If the EECONFIG pin is high during boot up the AT/LANTIC Controller's configuration is read from the EEPROM, before the PROM data is read. The configuration data is stored within the upper two words of the EEPROM's address space. Configuration Registers A and B are located in the lower of these words, Register C in the lower byte of the upper word, as shown in *Figure 10*.

To write this configuration into the EEPROM the user must follow the routine specified in the pseudo code below. This operation will work regardless of the level on EECONFIG. The EELOAD bit of Configuration Register B being set starts the EEPROM write process. Care should be taken not to accidentally set the GDLINK bit and therefore disable link integrity checking. The next 3 writes to this register load the values that will be stored in the configuration register (note that the last 2 of these writes do not have to follow the normal practice of preceding a write to this register with a read to this address). The AT/LANTIC Controller will then commence the EEPROM write. The write has been completed when the EELOAD bit goes to zero. This loading procedure should be followed exactly and interrupts should be disabled until it has completed, to prevent any accidental accesses to the AT/LANTIC Controller.

EEPROM\_LOAD()

```
{
    DISABLE_INTERRUPTS();
    value = READ(CONFIG_B);
    value = value AND 1 GDLINK;
    value = value OR EELOAD;
    WRITE(CONFIG_B, value);
    READ(CONFIG_B);
    WRITE(CONFIG_B, config_for_A);
    WRITE(CONFIG_B, config_for_B);
    WRITE(CONFIG_B, config_for_C);
    while (value AND EELOAD)
    {
        value = READ(CONFIG_B);
        WAIT();
    }
    ENABLE_INTERRUPTS();
}
```

### 4.4 JUMPERED AND JUMPERLESS OPERATION SUPPORT

The AT/LANTIC Controller supports several options that enable the implementation of either a "jumpered" or "jumperless" power on configuration when installed into a standard PC compatible's ISA bus. A wide range of options are provided to ensure that the AT/LANTIC Controller can be configured by an end user to function in all possible PC-AT system configurations. Several types of configuration options can be implemented examples including:

1. Full jumper options: All programmable options are selected by utilizing jumpers on the board. Option selection requires no special software. An example of this is shown in the *Figure 11*.

2. I/O address jumpers only: All other options configurable via software. This option simplifies installation while maximizing compatibility.
3. Jumperless: Special scheme provides contention-free I/O address selection.

The AT/LANTIC Controller's Configuration Registers are the key to providing the ability to implement various configuration options. These registers are configured by the same method in shared memory and I/O port modes, 8- or 16-bit modes. The bit definitions of these registers are provided in Section 5. All three registers are configured by hardware selection during the Power-On-Reset of the system. Two of these registers can be configured via software (the Mode Configuration Registers A and B). The third register (Hardware Configuration Register C) is only configured during reset.

The following table indicates *most* of the AT/LANTIC Controller options that a designer may like to have user configurable. (This list does not represent the complete list. For the full list see the Configuration register descriptions in Section 5.)

**TABLE I. Some Configuration Options for AT/LANTIC Controller**

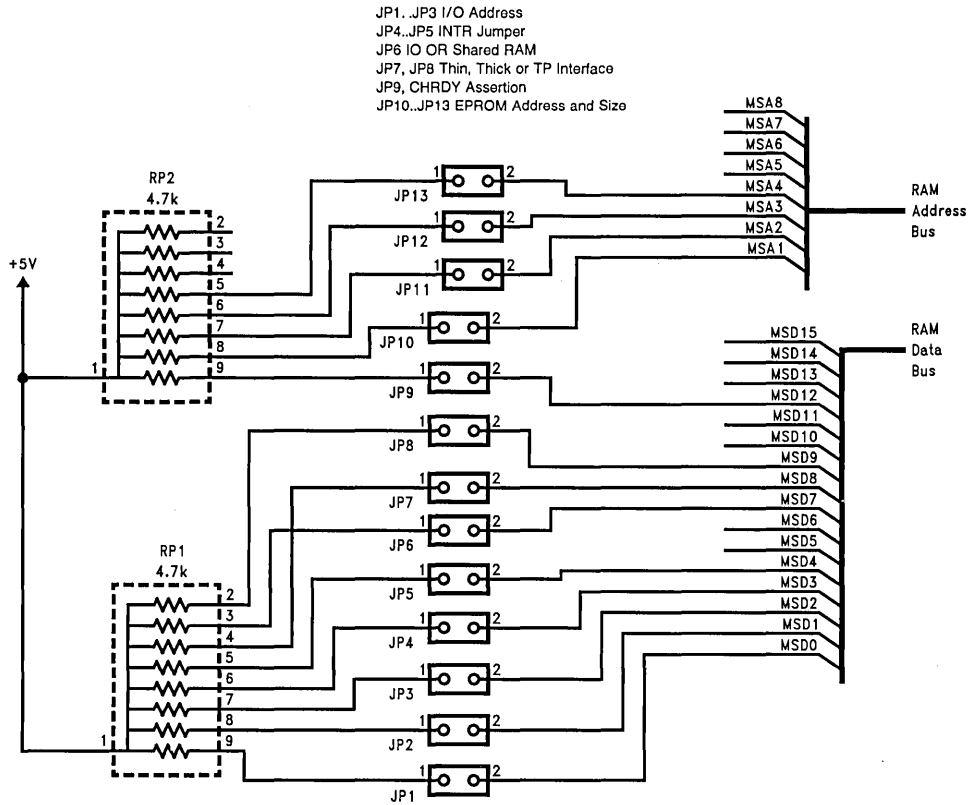
Option	Selections	
I/O Base Address	0300H Software 0240H 0280H	02C0H 0320H 0340H 0360H
Interrupt No.	4 Interrupts	8 Interrupts
Boot PROM Address	Disabled 0C000H 0C400H 0C800H	0CC00H 0D000H 0D400H 0D800H 0DC00H
Boot PROM Size	None 16k	32k 64k
Media Selection	Twisted Pair AUI Port	Thin Ethernet
Architecture	I/O Mode	Shared RAM Mode
Bus Timing Options	IOCHRDY Mode	MEM16 Mode

The three basic options are described below. Because of the variety of programmable options there are a number of variations possible, only a few typical examples will be discussed.

#### FULLY JUMPERED OPERATION

This option is shown in *Figure 11*. In this configuration most options are selected by jumpers on the AT/LANTIC Controller's memory bus. For this option all configuration options are set upon power-on by the AT/LANTIC Controller as described in Section 4.2. Accessing the configuration registers is unnecessary and the EEPROM need only contain the

## 4.0 Functional Description (Continued)



TL/F/11498-B

FIGURE 11. Example of Jumper Configuration

Ethernet ID address (Configuration Register B bit 7 should be set to disable EEPROM configuration mode, and Configuration Register C bit 7 could be set to disable software configuration completely).

### MINIMAL JUMPERS

The AT/LANTIC Controller's configuration registers provide the capability to enable software to configure various options (some may be hardwired). For the one option that is not easily configured on the ISA bus is the I/O address options. The reason for this is that the I/O locations must first be known by the software in order for the software (usually a device driver) to access the AT/LANTIC Controller. However, upon power up, in order to access a register to configure the I/O base address to avoid conflicts some default location must be given (typically set in hardware on the memory bus). It is possible that this default location conflicts with an already installed device. If this is the case then one possible solution, is to provide a jumper option for only the I/O Addresses. A similar situation exists for the boot PROM memory addresses.

In this application all options except the I/O address and the boot PROM are hardwired on the memory bus to a default setting. After power up software can change the con-

figuration to avoid conflicts on these settings. The advantage of this approach is that for most systems the default I/O address setting is the correct one and no installation will be required in this case. This approach minimizes any compatibility issues.

### NO JUMPERS

The conflicts possible in the I/O base selection can be overcome by a special mode for software configuration of the I/O base address. By using this mode, and by using the configuration storage capability of the EEPROM a fully software configurable design on the ISA bus can be realized without address conflict problems.

This mode is invoked by having the AT/LANTIC Controller default to jumperless software configuration option in the I/O base selection. This mode enables configuration register A to be mapped to address location 278H which is defined to be a printer port's data register. If software writes to this location four consecutive times on the fourth write the AT/LANTIC Controller will load the data written into the I/O address bits of Configuration Register A. This data should set the I/O base address to a known conflict-free value. The AT/LANTIC Controller can now be configured and operated

## 4.0 Functional Description (Continued)

at the desired base I/O address. If desired the configuration software could change the EEPROM content to the new values eliminating the need to reconfigure upon each power up. Alternately the software could leave the EEPROM alone and execute the configuration using the printer port's data register upon each power up. This configuration scheme will only work once after each power-up. Therefore the user cannot enable the AT/LANTIC Controller from reserved mode, change it back into reserved mode, and enable it again. A power-on reset must occur between the first time it is enabled from the reserved mode and the second.

A second consideration is the location of the boot PROM in the system memory map, which also has the same conflict and programming considerations as the I/O address selection. However the solution is different, primarily because the boot PROM must be configured before power up. This is because during normal usage of the boot PROM the PC's BIOS will look for the ROM immediately after reset, not allowing configuration software to first select the boot PROM addressing prior to usage.

To configure the boot PROM without jumpers the configuration software must first power up the AT/LANTIC Controller, configure the EEPROM to the desired location, then hardware reset the AT/LANTIC Controller. After the reset the AT/LANTIC Controller's EEPROM will load in the desired boot PROM configuration automatically during the reset. Now after reset when the PC scans for the boot PROM, the ROM will be correctly mapped in the memory space enabling the network boot operation to proceed.

### Ethernet Cable Configuration

AT/LANTIC Controller offers the choice of all the possible Ethernet cabling options, that is Ethernet (10BASE5), Thin Ethernet (10BASE2) and Twisted-pair Ethernet (10BASE-T). The type of cabling used is controlled by Configuration Register B. AT/LANTIC Controller also supplies a THIN output signal which can be used to disable/enable an external DC-DC converter which is required for 10BASE2.

### 4.5 LOW POWER OPERATION

The AT/LANTIC Controller has a low power support mode that can be used to disable the Ethernet port and conserve power. It should be noted that the device is not operational in this mode and requires to be initialized after exiting this mode.

The power and ground pins to the AT/LANTIC Controller are split up into two groups, interface and core. By switching the power off to the core logic while still powering the interface logic the AT/LANTIC Controller can be powered down without crashing the ISA bus. The LOWPWR pin should be driven high to indicate that the device is about to go into low power then the power to the  $V_{DD}$  pins should be switched off. The same signal that is used to drive the LOWPWR pin can be used to drive a p-channel load switch to disable power to the core. This switch must have a very low on resistance to minimize the voltage difference between the  $V_{CC}$  and the  $IFV_{CC}$ . All devices on the memory support bus should also be powered from the  $V_{CC}$  supply.

### 4.6 BOOT PROM OPERATION

The AT/LANTIC Controller supports an optional boot PROM, the address and size of which can be set in Configuration Register C. This boot PROM can be any 8 bits wide storage device implemented with a non-volatile technology. Write cycles to this device can be enabled and disabled by programming Configuration Register B. This can be used to prevent unwanted write cycles to certain devices, such as a Flash EEPROM. **It should be noted that the address pins for the boot PROM should be connected directly to the ISA bus.** The AT/LANTIC Controller supplies the chip select to the device and buffers the data onto and from the ISA bus, so the memory support data bus should be connected to the boot PROM's data pins.

### 4.7 DP8390 CORE (NETWORK INTERFACE CONTROLLER)

The DP8390 Core logic, *Figure 12*, contains the Serializer/Deserializer which is controlled by the Protocol PLA, DMA Control, FIFO, Address Comparator, Multicast Hashing Register. The DP8390 core implements all of the IEEE 802.3 Media access control functions for the AT/LANTIC Controller, and interfaces to the internal ENDEC (on the left of the block diagram) and also interfaces to the Bus Interface and memory support bus via a number of address, data and control signal (and the right side of the block diagram). The following sections describe the functions of the DP8390 core.

#### Receive Deserializer

The Receive Deserializer is activated when the input signal Carrier Sense is asserted to allow incoming bits to be shifted into the shift register by the receive clock. The serial receive data is also routed to the CRC generator/checker. The Receive Deserializer includes a synch detector which detects the SFD (Start of Frame Delimiter) to establish where byte boundaries within the serial bit stream are located. After every eight receive clocks, the byte wide data is transferred to the 16-byte FIFO and the Receive Byte Count is incremented. The first six bytes after the SFD are checked for valid comparison by the Address Recognition Logic. If the Address Recognition Logic does not recognize the packet, the FIFO is cleared.

#### CRC Generator/Checker

During transmission, the CRC logic generates a local CRC field for the transmitted bit sequence. The CRC encodes all fields after the synch byte. The CRC is shifted out MSB first following the last transmit byte. During reception the CRC logic generates a CRC field from the incoming packet. This local CRC is serially compared to the incoming CRC appended to the end of the packet by the transmitting node. If the local and received CRC match, a specific pattern will be generated and decoded to indicate no data errors. Transmission errors result in different patterns and are detected, resulting in rejection of a packet.

## 4.0 Functional Description (Continued)

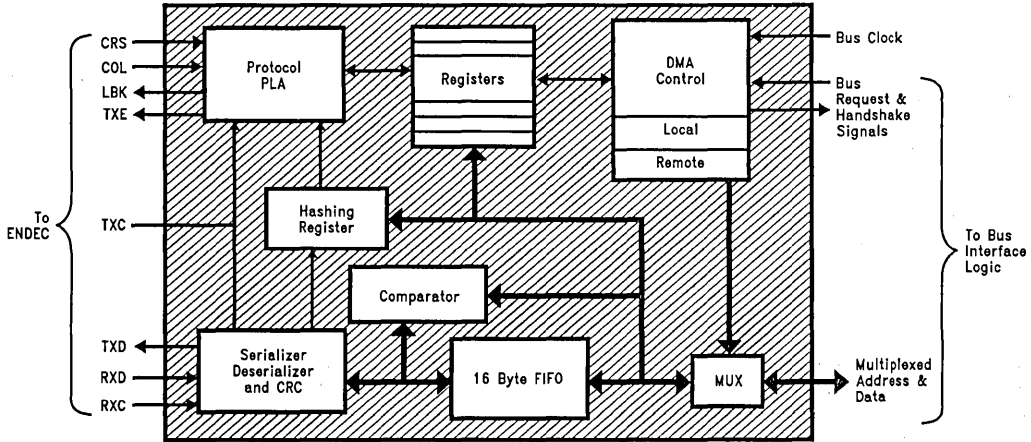


FIGURE 12. DP8390 Controller Core Simplified Block Diagram

TL/F/11498-9

### Transmit Serializer

The Transmit Serializer reads parallel data from the FIFO and serializes it for transmission. The serializer is clocked by the transmit clock generated internally. The serial data is also shifted into the CRC generator/checker. At the beginning of each transmission, the Preamble and Synch Generator append 62 bits of 1,0 preamble and a 1,1 synch pattern. After the last data byte of the packet has been serialized the 32-bit FCS (Frame Check Sequence) field is shifted directly out of the CRC generator. In the event of a collision the Preamble and Synch generator is used to generate a 32-bit JAM pattern of all 1's.

### Comparator-address Recognition Logic

The address recognition logic compares the Destination Address Field (first 6 bytes of the received packet) to the Physical address registers stored in the Address Register Array. If any one of the six bytes does not match the pre-programmed physical address, the Protocol Control Logic rejects the packet. All multicast destination addresses are filtered using a hashing technique. (See register description.) If the multicast address indexes a bit that has been set in the filter bit array of the Multicast Address Register Array the packet is accepted, otherwise it is rejected by the Protocol Control Logic. Each destination address is also checked for all 1's which is the reserved broadcast address.

### FIFO and Packet Data Operations

#### OVERVIEW

To accommodate the different rates at which data comes from (or goes to) the network and goes to (or comes from) the packet buffer memory, the AT/LANTIC Controller contains a 16-byte FIFO for buffering data between the media and the buffer RAM located on the memory support bus. The FIFO threshold is programmable, allowing filling (or emptying) the FIFO at different rates. When the FIFO has filled to its programmed threshold, the local DMA channel transfers these bytes (or words) into local memory (via the

memory bus). It is crucial that the local DMA is given access to the bus within a minimum bus latency time; otherwise a FIFO underrun (or overrun) occurs.

FIFO underruns or overruns are caused when a local DMA request is issued while an ISA bus access is current and the ISA cycle takes longer to complete than the local DMA's tolerable latency. This tolerable latency depends on the FIFO threshold, whether it is in byte or word wide mode and the speed of the DMA clock (BSCLK frequency). Note that this refers to standard ISA cycles NOT those where the CHRDY is deasserted extending the cycle.

#### FIFO THRESHOLD DETECTION

To assure that there is no overwriting of data in the FIFO, the FIFO logic flags a FIFO overrun as the 13th byte is written into the FIFO, effectively shortening the FIFO to 13 bytes. The FIFO logic also operates differently in Byte Mode and in Word Mode. In Byte Mode, a threshold is indicated when the n+1 byte has entered the FIFO; thus, with an 8 byte threshold, the AT/LANTIC Controller issues a request to the buffer RAM when the 9th byte has entered the FIFO, making the effective threshold 9 bytes. For Word Mode, the request is not generated until the n+2 bytes have entered the FIFO. Thus, with a 4 word threshold (equivalent to 8 byte threshold), a request to the buffer RAM is issued when the 10th byte has entered the FIFO, making the effective threshold 10 bytes.

#### TOLERABLE LATENCY CALCULATION

To prevent a FIFO **overrun** a byte (or word) of data must be **removed** from the FIFO before the 13th byte is written. Therefore the worst case tolerable latency is the time from the effective threshold being reached to the time the 13th byte is written minus the time taken to load the first byte (or word) of data to the FIFO during a local DMA burst (8 BSCLKs).

$$\begin{aligned} \text{tolerable latency} = & ((\text{overrun} - \text{effective}) \text{ threshold} \\ & \times \text{time to transfer byte on network}) \\ & - \text{time to fill 1st FIFO location} \end{aligned}$$

1

## 4.0 Functional Description (Continued)

For the case of a 4 word threshold using a 20 MHz BSCLK:

$$\begin{aligned} \text{tolerable latency} &= ((13 - 10) \times 800) - (8 \times 50) \text{ ns} \\ &= 2 \mu\text{s} \end{aligned}$$

To prevent a FIFO **underrun** a byte (or word) of data must be **added** from the FIFO before the last byte is removed. Therefore the worst case tolerable latency is the time from the effective threshold being reached to the time the last byte is removed minus the time taken to load the first byte (or word) of data to the FIFO during a local DMA burst (8 BSCLKs).

$$\begin{aligned} \text{tolerable latency} &= (\text{threshold} \\ &\quad \times \text{time to transfer byte on network}) \\ &\quad - \text{time to fill 1st FIFO location} \end{aligned}$$

For the case of a 4 word threshold using a 20 MHz BSCLK:

$$\begin{aligned} \text{tolerable latency} &= (4 \times 800) - (8 \times 50) \text{ ns} \\ &= 2.8 \mu\text{s} \end{aligned}$$

The worst case latency, either overrun or underrun, ultimately limits the overall latency that the AT/LANTIC Controller can tolerate. If the standard ISA cycles are shorter than the worst case latency then no FIFO overruns or underruns will occur.

### BEGINNING OF RECEIVE

At the beginning or reception, the AT/LANTIC Controller stores entire Address field of each incoming packet in the FIFO to determine whether the packet matches its Physical Address Registers or maps to one of its Multicast Registers. This causes the FIFO to accumulate 8 bytes.

Furthermore, there are some synchronization delays in the DMA PLA. Thus, the actual time that a request to access the buffer RAM is asserted from the time the Start of Frame Delimiter (SFD) is detected is 7.8  $\mu\text{s}$ . This operation affects the bus latencies at 2 byte and 4 byte thresholds during the first receive request since the FIFO must be filled to 8 bytes (or 4 words) before issuing a request to the buffer RAM.

### END OF RECEIVE

When the end of a packet is detected by the ENDEC module, the AT/LANTIC Controller enters its end of packet processing sequence, emptying its FIFO and writing the status information at the beginning of the packet. The AT/LANTIC Controller holds onto the memory bus for the entire sequence. The longest time that local DMA will hold the buffer RAM occurs when a packet ends just as the AT/LANTIC Controller performs its last FIFO burst. The AT/LANTIC Controller, in this case, performs a programmed burst transfer followed by flushing the remaining bytes in the FIFO, and completed by writing the header information to the buffer memory. The following steps occur during this sequence.

1. AT/LANTIC Controller issues request to access the RAM because the FIFO threshold has been reached.
2. During the burst, packet ends, resulting in the request being extended.
3. AT/LANTIC Controller flushes remaining bytes from FIFO.
4. AT/LANTIC Controller performs internal processing to prepare for writing the header.
5. AT/LANTIC Controller writes 4-byte (2-word) header
6. AT/LANTIC Controller de-asserts access to the buffer RAM.

### BEGINNING OF TRANSMIT

Before transmitting, the AT/LANTIC Controller performs a prefetch from memory to load the FIFO. The number of bytes prefetched is the programmed FIFO threshold. The next request to the buffer RAM is not issued until after the AT/LANTIC Controller actually begins transmitting data, i.e., after SFD.

### READING THE FIFO

If the FIFO is read during normal operation the AT/LANTIC Controller will "hang" the ISA bus by deasserting CHRDY and never asserting it. The FIFO should only be read during loopback diagnostics, when it will operate normally.

### PROTOCOL PLA

The Protocol PLA is responsible for implementing the IEEE 802.3 protocol, including collision recovery with random backoff. The Protocol PLA also formats packets during transmission and strips preamble and synch during reception.

### DMA AND BUFFER CONTROL LOGIC

The DMA and Buffer Control Logic is used to control two 16-bit DMA channels. During reception, the Local DMA stores packets in a receive buffer ring, located in buffer memory. During transmission the Local DMA uses programmed pointer and length registers to transfer a packet from local buffer memory to the FIFO.

A second DMA channel is used when the AT/LANTIC Controller is used in I/O Port mode. This DMA is used as a slave DMA to transfer data between the local buffer memory and the host system. The Local DMA and Remote DMA are internally arbitrated, with the Local DMA channel having highest priority. Both DMA channels use a common external bus clock to generate all required bus timing. External arbitration is performed with a standard bus request, bus acknowledge handshake protocol.

In the shared memory mode the Remote DMA is not used, because in this mode the system has direct read/write access to the buffer RAM.

## 4.0 Functional Description (Continued)

### 4.8 TWISTED PAIR INTERFACE MODULE

The TPI consists of five main logical functions:

- The Receiver/Smart Squelch, responsible for determining when valid data is present on the differential receive inputs (RXI±) and receiving the data.
- The Collision function checks for simultaneous transmission and reception of data on the TXO± and RXI± pins.
- The Link Detector/Generator checks the integrity of the cable connecting the two twisted pair MAUs.
- The Jabber disables the transmitter if it attempts to transmit a longer than legal packet.
- The TX Driver and Pre-emphasis transmits Manchester encoded data to the twisted pair network via the summing resistors and transformer/filter.

### Receiver and Smart Squelch

The AT/LANTIC Controller implements an intelligent receive squelch on the RXI± differential inputs to ensure that impulse noise on the receive inputs will not be mistaken for a valid signal.

The squelch circuitry employs a combination of amplitude and timing measurements to determine the validity of data on the twisted pair inputs. There are two voltage level options for the smart squelch. One mode, 10BASE-T mode, uses levels that meet the 10BASE-T specification. The second mode, reduced squelch mode, uses a lower squelch threshold level, and can be used in longer cable applications where smaller signal levels may be applied. The squelch level mode can be selected in the AT/LANTIC Controller configuration registers.

Figure 14 shows the operation of the smart squelch in 10BASE-T mode.

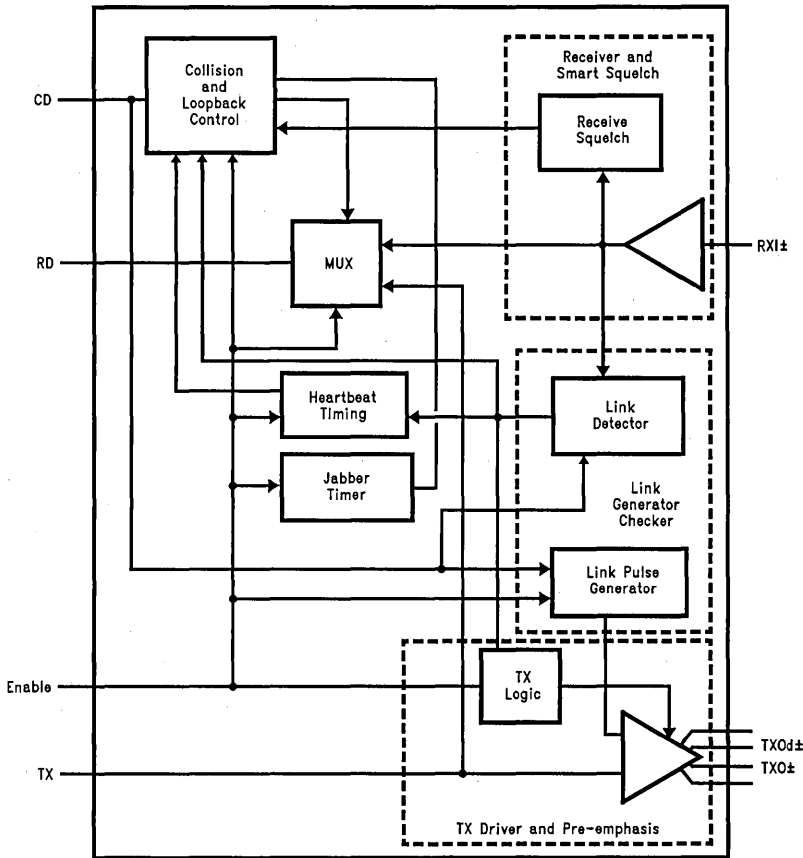


FIGURE 13. Twisted Pair Interface Module Block Diagram

TL/F/11498-10

## 4.0 Functional Description (Continued)

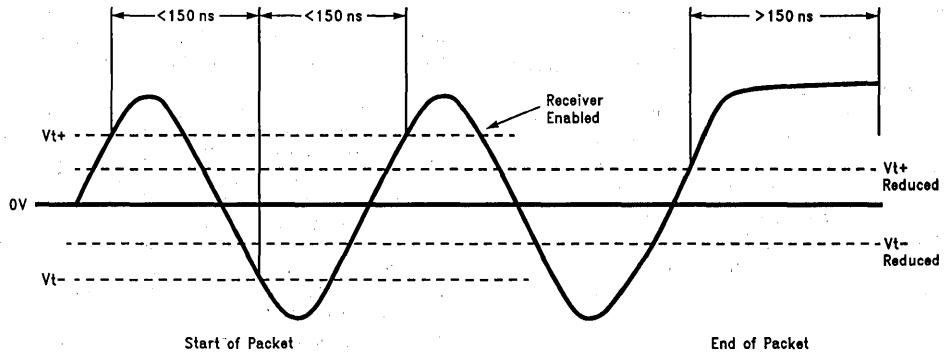


FIGURE 14. Twisted Pair Squelch Waveform

TL/F/11498-11

The signal at the start of packet is checked by the smart squelch and any pulses not exceeding the squelch level (either positive or negative, depending upon polarity) will be rejected. Once this first squelch level is overcome correctly the opposite squelch level must then be exceeded within 150 ns later. Finally the signal must exceed the original squelch level within a further 150 ns to ensure that the input waveform will not be rejected. The checking procedure results in the loss of typically three bits at the beginning of each packet.

Only after all these conditions have been satisfied will a control signal be generated to indicate to the remainder of the circuitry that valid data is present. At this time the smart squelch circuitry is reset.

In the reduced squelch mode the operation is identical except that the lower squelch levels shown in *Figure 14* are used.

Valid data is considered to be present until either squelch level has not been generated for a time longer than 150 ns, indicating End of Packet. Once good data has been detected the squelch levels are reduced to minimize the effect of noise causing premature End of Packet detection.

### Collision

A collision is detected by the TPI module when the receive and transmit channels are active simultaneously. If the TPI is receiving when a collision is detected it is reported to the controller immediately. If, however, the TPI is transmitting when a collision is detected the collision is not reported until seven bits have been received while in the collision state. This prevents a collision being reported incorrectly due to noise on the network. The signal to the controller remains for the duration of the collision.

Approximately 1  $\mu\text{s}$  after the transmission of each packet a signal called the Signal Quality Error (SQE) consisting of typically 10 cycles of 10 MHz is generated. This 10 MHz signal, also called the Heartbeat, ensures the continued functioning of the collision circuitry.

### Link Detector/Generator

The link generator is a timer circuit that generates a link pulse as defined by the 10 Base-T specification that will be generated by the transmitter section. The pulse which is 100 ns wide is transmitted on the TXO+ output, every 16 ms, in the absence of transmit data.

The pulse is used to check the integrity of the connection to the remote MAU. The link detection circuit checks for valid pulses from the remote MAU and if valid link pulses are not received the link detector will disable the transmit, receive and collision detection functions.

The  $\overline{\text{GDLNK}}$  output can directly drive a LED to show that there is a good twisted pair link. For normal conditions the LED will be on. The link integrity function can be disabled by setting the GDLNK bit of Configuration Register B.

### Jabber

The jabber timer monitors the transmitter and disables the transmission if the transmitter is active for greater than 26 ms. The transmitter is then disabled for the whole time that the Endec module's internal transmit enable is asserted. This signal has to be deasserted for approximately 750 ms (the unjab time) before the Jabber re-enables the transmit outputs.

### Transmitter

The transmitter consists of four signals, the true and complement Manchester encoded data (TXO $\pm$ ) and these signals delayed by 50 ns (TXOd $\pm$ )

These four signals are resistively combined TXO+ with TXOd- and TXO- with TXOd+. This is known as digital pre-emphasis and is required to compensate for the twisted pair cable which acts like a low pass filter causing greater attenuation to the 10 MHz (50 ns) pulses of the Manchester encoded waveform than the 5 MHz (100 ns) pulses.

An example of how these signals are combined is shown in the following diagram.



### 4.0 Functional Description (Continued)

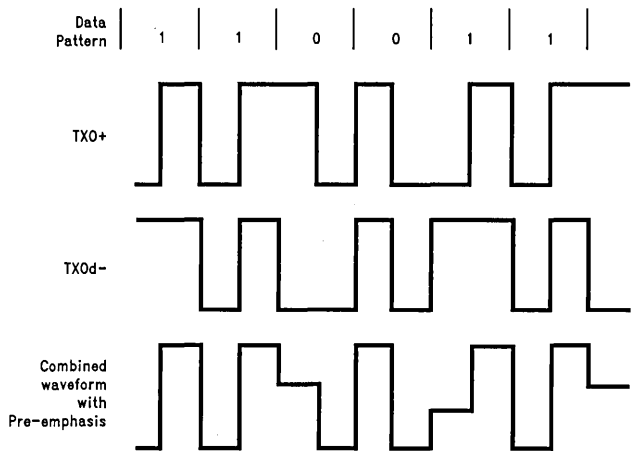


FIGURE 15. Typical Summed Transmit Waveform

TL/F/11498-12

The signal with pre-emphasis shown above is generated by resistively combining TX0+ and TX0d-. This signal along with its complement is passed to the transmit filter.

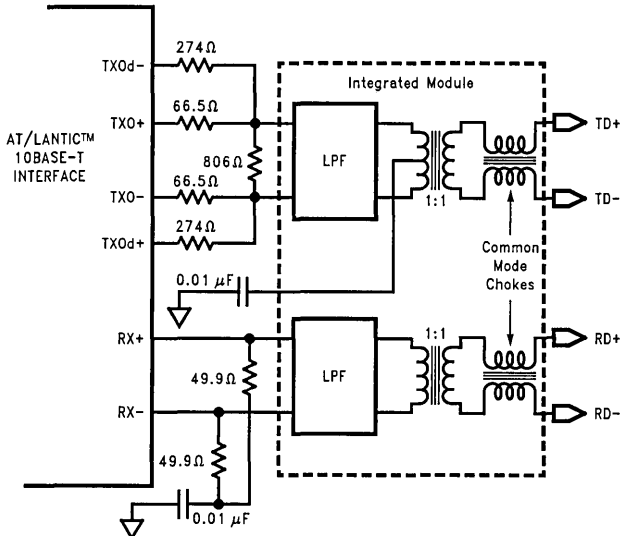


FIGURE 16. External Circuitry to Connect AT/LANTIC Controller to Twisted Pair Cable

TL/F/11498-13

### 4.0 Functional Description (Continued)

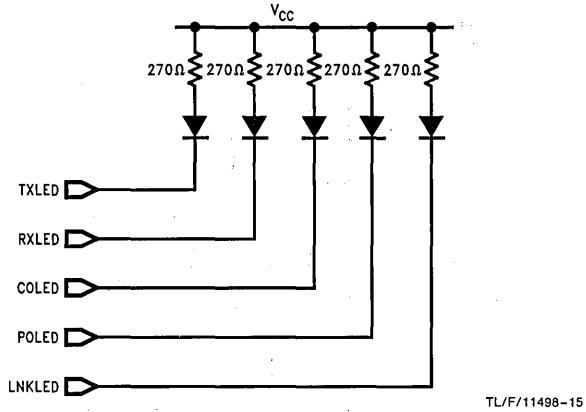


FIGURE 17. Typical AT/LANTIC Controller LED Connection

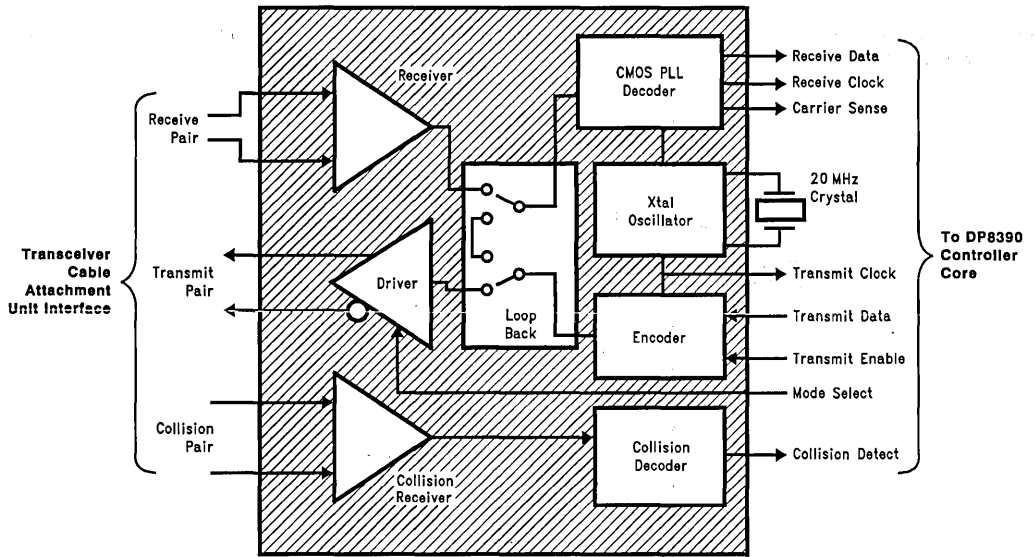


FIGURE 18. Encoder/Decode Block Diagram

## 4.0 Functional Description (Continued)

### Status Information

Status information is provided by the AT/LANTIC Controller on the  $\overline{RXLED}$ ,  $\overline{TXLED}$ ,  $\overline{COLED}$  and  $\overline{POL}$  outputs as described in the pin description table. These outputs are suitable for driving status LED's as shown in *Figure 17*. All outputs are open drain.

Recommended integrated Filter-Transformer-choke modules:

1. Pulse Engineering PE65424
2. Valor FL1012 or FL1030.

### 4.9 ENCODER/DECODER (ENDEC) MODULE

The ENDEC consists of four main logical blocks:

- a. The oscillator generates the 10 MHz transmit clock signal for system timing.
- b. The Manchester encoder accepts NRZ data from the controller, encodes the data to Manchester, and transmits the data differentially to the transceiver, through the differential transmit driver.
- c. The Manchester decoder receives Manchester data from the transceiver, converts it to NRZ data and clock pulses, and sends it to the controller.
- d. The collision translator indicates to the controller the presence of a valid 10 MHz collision signal to the PLL.

### Oscillator

The oscillator is controlled by a 20 MHz parallel resonant crystal connected between X1 and X2 or by an external clock on X1. The 20 MHz output of the oscillator is divided by 2 to generate the 10 MHz transmit clock for the controller. The oscillator also provides internal clock signals to the encoding and decoding circuits.

### CRYSTAL OPERATION

If the crystal used with the internal oscillator circuit is not properly selected, the AT/LANTIC Controller oscillator may not reliably start oscillation under all conditions.

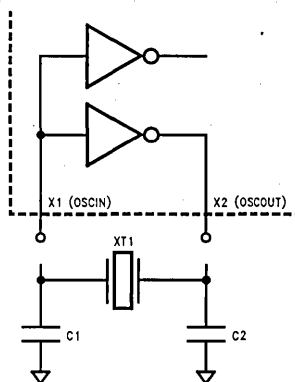
If this occurs, it could be deceiving to a designer, since his prototypes may work fine. However, when the designer does qualification testing or starts production, he may encounter a higher than expected board yield loss due to the oscillator not starting. The AT/LANTIC Controller's oscillator circuit clocks the Encoder-Decoder logic. The AT/LANTIC Controller's oscillator also clocks the twisted pair interface block. If the oscillator does not start, the AT/LANTIC Controller will not be able to transmit or receive.

If a crystal is connected to the AT/LANTIC Controller, it is recommended that the circuit shown in *Figure 19* be used and that the components used meet the following:

- Crystal XT1: AT cut parallel resonant crystal  
 Series Resistance:  $\leq 25\Omega$   
 Specified Load Capacitance:  $\leq 20$  pF  
 Accuracy: 0.005% (50 ppm)  
 Typical Load: 50  $\mu$ W–75  $\mu$ W

The recommended values for capacitors C1 and C2 are 26 pF minus the board capacitance on that pin. Therefore if both X1 and X2 have 4 pF of board capacitance, then a 22 pF capacitor should be used.

According to the IEEE 802.3 standard, the entire oscillator circuit (crystal and amplifier) must be accurate to 0.01%. When using a crystal, the X2 pin is not guaranteed to provide a TTL compatible logic output, and should not be used



TL/F/11498-16

**FIGURE 19. Crystal Connection to AT/LANTIC Controller (see text for component values)**

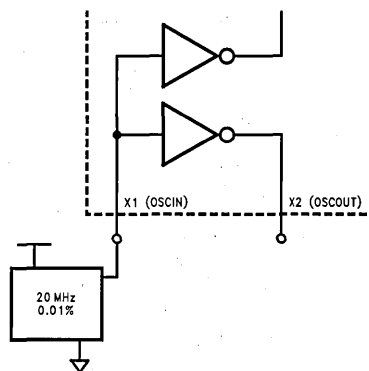
to drive external standard logic. If additional logic needs to be driven, then an external oscillator should be used, as described in the following section.

### OSCILLATOR MODULE OPERATION

If the designer wishes to use a crystal clock oscillator, one that provides the following should be employed:

1. TTL or CMOS output with a 0.01% frequency tolerance
2. 40%–60% duty cycle

The circuit is shown in *Figure 20*. When using a clock oscillator it is recommended that the designer connect the oscillator output to the X1 pin and leave the X2 pin floating.



TL/F/11498-17

**FIGURE 20. AT/LANTIC Controller Connection for Oscillator Module**

### Manchester Encoder and Differential Driver

The differential transmit pair, on the secondary of the employed transformer, drives up to 50 meters of twisted pair AUI cable. These outputs are source followers which require two 270 $\Omega$  pull-down resistors to ground (see *Figure 21*).

The AT/LANTIC Controller allows full-step to be compatible with IEEE 802.3. Transmit+ and Transmit- are equal in the idle state, providing zero differential voltage to operate with transformer coupled loads.

### Manchester Decoder

The decoder consists of a differential receiver and a PLL to separate a Manchester encoded data stream into internal

## 4.0 Functional Description (Continued)

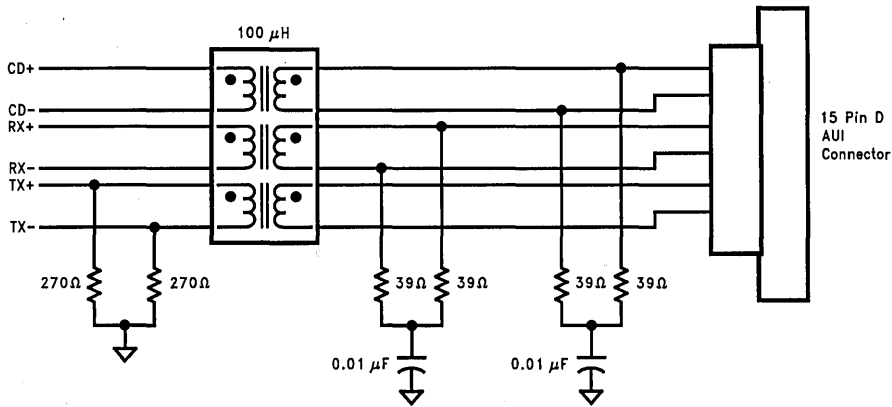


FIGURE 21. Connection from AT/LANTIC Controller's AUI Port to the AUI Connector

TL/F/11498-19

clock signals and data. The differential input must be externally terminated with two 39Ω resistors connected in series if the standard 78Ω transceiver drop cable is used, in thin Ethernet applications, these resistors are optional. To prevent noise from falsely triggering the decoder, a squelch circuit at the input rejects signals with levels less than -175 mV. Data becomes valid typically within 6 bit times. The AT/LANTIC Controller may tolerate bit jitter up to 20 ns in the received data. The decoder detects the end of a frame when no more mid-bit transitions are detected.

### Collision Translator

When in AUI Mode, the Ethernet transceiver (DP8392 CT) detects a collision, it generates a 10 MHz signal to the differential collision inputs (CD±) of the AT/LANTIC Controller. When these inputs are detected active, the AT/LANTIC Controller uses this signal to back off its current transmission and reschedule another one.

In this mode the  $\overline{\text{COLED}}$  output will indicate when the CD± lines are active during activity on the network. This means it will correctly indicate any collision on the network, but will not be lit for heartbeat or if there is no cable connected.

The collision differential inputs are terminated the same way as the differential receive inputs. The squelch circuitry is also similar, rejecting pulse levels less than -175 mV.

### PLL V<sub>CC</sub> Power Supply Consideration

The PLL V<sub>CC</sub> pin is the +5V power supply for the phase lock loop (PLL) of the ST-NIC ENDEC unit. Since this is an

analog circuit, excessive noise on the PLL V<sub>CC</sub> pin can affect the performance of the PLL. This noise, if in the 10 kHz-400 kHz range, can reduce the jitter performance of the ENDEC, resulting in missing packets or CRC errors.

If the power supply noise is causing significant packet reception error, a low pass filter could be added to reduce the power supply noise and hence improve the jitter performance. Standard analog design techniques should be utilized when laying out the power supply traces on the board. If the digital power supply is used, it may be desirable to add a one pole RC filter (designed to have a cut-off frequency of 1 kHz) as shown in Figure 4 to improve the jitter performance. The PLL V<sub>CC</sub> only draws 3 mA-4 mA so the voltage across the resistor is less than 90 mV, which will not affect the PLL's operation.

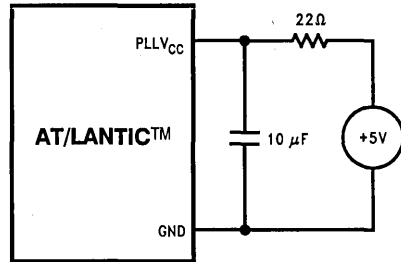


FIGURE 22. Filtering Power Supply Noise

TL/F/11498-18

## 5.0 Register Descriptions

### 5.1 CONFIGURATION REGISTERS

These registers are used to configure the operation of the AT/LANTIC Controller typically after power up. These registers control the configuration of bus interface, setting options like interrupt selection, I/O base address, and other specific modes.

#### MODE CONFIGURATION REGISTER A

To prevent any accidental writes of this register it is "hidden" behind a previously unused register. Register 0AH in the AT/LANTIC Controller's Page 0 of registers was previously reserved on a read. Now Configuration Register A can be read at that address and can be written to by following a read to 0AH with a write to 0AH. If any other AT/LANTIC Controller register accesses take place between the read and the write then the write to 0AH will access the Remote Byte Count Register 0.

7	6	5	4	3	2	1	0
MEMIO	FREAD	INT2	INT1	INT0	IOAD2	IOAD1	IOAD0

Bits	Symbols	Function																				
0-2	IOAD0- IOAD2	<p><b>I/O ADDRESS:</b> These three bits determine the base I/O address of the AT/LANTIC Controller, within the system's I/O map. The AT/LANTIC Controller occupies 20H bytes of the system's address space.</p> <table style="margin-left: 20px;"> <tr><td>0 0 0</td><td>0300H</td></tr> <tr><td>0 0 1</td><td>Software (Note 1)</td></tr> <tr><td>0 1 0</td><td>0240H</td></tr> <tr><td>0 1 1</td><td>0280H</td></tr> <tr><td>1 0 0</td><td>02C0H</td></tr> <tr><td>1 0 1</td><td>0320H</td></tr> <tr><td>1 1 0</td><td>0340H</td></tr> <tr><td>1 1 1</td><td>0360H</td></tr> </table> <p><b>Note 1:</b> When 001 is selected the AT/LANTIC controller will not respond to any I/O Addresses, but will allow 4 consecutive writes to 278H to write these three bits of this register. This sequence will only operate once after a power-on reset. This mode allows the AT/LANTIC Controller to be configured via software without conflicting with other peripherals.</p>	0 0 0	0300H	0 0 1	Software (Note 1)	0 1 0	0240H	0 1 1	0280H	1 0 0	02C0H	1 0 1	0320H	1 1 0	0340H	1 1 1	0360H				
0 0 0	0300H																					
0 0 1	Software (Note 1)																					
0 1 0	0240H																					
0 1 1	0280H																					
1 0 0	02C0H																					
1 0 1	0320H																					
1 1 0	0340H																					
1 1 1	0360H																					
3-5	INT0- INT2	<p><b>INTERRUPT LINE USED:</b> There are two interrupt modes which can be enabled by setting bit 5 of Configuration Register C to the appropriate level.</p> <p><b>DIRECT DRIVE MODE:</b> In this mode an interrupt output pin will be driven active on a valid interrupt condition. Only one pin may be driven, the other three will remain at TRI-STATE®. The pin driven is determined by the value in this register.</p> <table style="margin-left: 20px;"> <thead> <tr> <th>Bit 5</th> <th>Bit 4</th> <th>Bit 3</th> <th>Interrupt</th> </tr> </thead> <tbody> <tr><td>X</td><td>0</td><td>0</td><td>INT0</td></tr> <tr><td>X</td><td>0</td><td>1</td><td>INT1</td></tr> <tr><td>X</td><td>1</td><td>0</td><td>INT2</td></tr> <tr><td>X</td><td>1</td><td>1</td><td>INT3</td></tr> </tbody> </table> <p><b>CODED OUTPUT MODE:</b> In this mode INT3 is the active interrupt output while pins INT0 to INT2 are programmable outputs reflecting the values on bits 3 to 5.</p>	Bit 5	Bit 4	Bit 3	Interrupt	X	0	0	INT0	X	0	1	INT1	X	1	0	INT2	X	1	1	INT3
Bit 5	Bit 4	Bit 3	Interrupt																			
X	0	0	INT0																			
X	0	1	INT1																			
X	1	0	INT2																			
X	1	1	INT3																			
6	FREAD	<p><b>FAST READ:</b> When this bit is set high the AT/LANTIC Controller, in I/O mode, will begin the next port fetch before the current IORD has completed. In slow ISA systems this may cause the data in the port to be overwritten before the ISA cycle has been completed.</p>																				
7	MEMIO	<p><b>MEMORY OR I/O MODE:</b> If this bit is set high then the AT/LANTIC Controller is in shared memory mode. If it is set low it is in I/O mode.</p>																				

## 5.0 Register Descriptions (Continued)

### Mode Configuration Register B

To prevent any accidental writes of this register it is "hidden" behind a previously unused register. Register 0BH in the AT/LANTIC Controller's Page 0 of registers was previously reserved on a read. Now Configuration Register B can be read at that address and can be written to by following a read to 0BH with a write to 0BH. If any other AT/LANTIC Controller register accesses take place between the read and the write then the write to 0BH will access the Remote Byte Count Register 1. **Care should be taken when writing to this register as GDLINK and BE are not simple read/write bits, e.g., the user cannot change the physical layer by reading B, or-ing the returned value with the bits to be set, and writing this value to B. This could inadvertently disable link integrity generation and clear a bus error indication before it was noted.**

7	6	5	4	3	2	1	0
EELoad	BPWR	BE	CHRDY	IO16CON	GDLINK	PHYS1	PHYS0

Bits	Symbols	Function
0-1	PHYS0- PHYS1	<b>PHYSICAL LAYER INTERFACE:</b> These 2 bits determine which type of physical interface the AT/LANTIC Controller is using. The 2 TPI interfaces use twisted pair outputs and inputs, while the other 2 interfaces use the AUI outputs and inputs. In 10BASE5 mode the THIN output pin is driven low, in 10BASE2 mode it is driven high. This can be used to enable the DC-DC converter required by the 10BASE2 specification to provide electrical isolation. The Non spec TPI mode is a twisted pair mode with reduced receive squelch levels. This allows the use of longer cable lengths than specified in the twisted pair specification, or use of cable with higher losses. <ul style="list-style-type: none"> <li>0 0 TPI (10BASE-T Compatible Squelch Level)</li> <li>0 1 Thin Ethernet (10BASE2)</li> <li>1 0 Thick Ethernet (10BASE5) (AUI Port)</li> <li>1 1 TPI (Reduced Squelch Level)</li> </ul>
2	GDLNK	<b>GOOD LINK:</b> When a 1 is written to this bit the link test pulse generation and integrity checking is disabled. When this bit is read it will indicate link status, reflecting the value shown on the LED output. It is 0 if the AT/LANTIC Controller is in AUI mode or if link testing is enabled and the link integrity is bad (i.e., the twisted pair link has been broken). It is 1 if the AT/LANTIC Controller is in TPI mode, link integrity checking is enabled and the link integrity is good (i.e., the twisted pair link has <b>not</b> been broken) or if the link testing is disabled.
3	IO16CON	<b>IO16 CONTROL:</b> When this bit is set high the AT/LANTIC Controller generates $\overline{IO16}$ after $\overline{IORD}$ or $\overline{IOWR}$ go active. If low this output is generated only on address decode.
4	CHRDY	<b>CHRDY FROM IORD OR IOWR OR FROM BALE:</b> When this bit is low the AT/LANTIC Controller will generate CHRDY after the command strobe. When high it will generate it after BALE goes high.
5	BE	<b>BUS ERROR:</b> This bit shows that the AT/LANTIC Controller has detected a bus error condition. This will go high if the AT/LANTIC Controller attempts to insert wait states into a system access and the system terminates the cycle without inserting the wait states. Writing a one to this bit clears it to zero. Writing a zero has no effect.
6	BPWR	<b>BOOT PROM WRITE:</b> When this bit is low no write cycles are generated to the boot PROM.
7	EELoad	<b>EEPROM LOAD:</b> Writing a 1 to this bit enables the EEPROM load algorithm as detailed in Section 4. <b>This bit should not be configured to be high, either from switches or an EEPROM.</b>

## 5.0 Register Descriptions (Continued)

### Hardware Configuration Register C

This register is configured during a RESET and can not be accessed by software.

7	6	5	4	3	2	1	0
SOFEN	CLKSEL	INTMOD	COMP	BPS3	BPS2	BPS1	BPS0

Bits	Symbols	Function																																																																																																
0-3	BPS0-3	<p><b>BOOT PROM SELECT:</b> Selects address at which boot PROM begins and the size. When the system reads within the selected memory area AT/LANTIC Controller reads the data in through MSD0-7 and drives it onto the system data bus. The following are valid addresses and sizes:</p> <table border="1" style="width: 100%; border-collapse: collapse; margin-top: 5px;"> <thead> <tr> <th style="width: 5%;">Bit 3</th> <th style="width: 5%;">Bit 2</th> <th style="width: 5%;">Bit 1</th> <th style="width: 5%;">Bit 0</th> <th style="width: 20%;">Address</th> <th style="width: 40%;">Size (I/O / Shared Mem.)</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>X</td> <td>X</td> <td>No boot PROM</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>0</td> <td>0C000H</td> <td>8k/16k</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>1</td> <td>0C400H</td> <td>8k/16k</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>0</td> <td>0C800H</td> <td>8k/16k</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>1</td> <td>0CC00H</td> <td>8k/16k</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>0</td> <td>0D000H</td> <td>8k/16k</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>1</td> <td>0D400H</td> <td>8k/16k</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>0</td> <td>0D800H</td> <td>8k/16k</td> </tr> <tr> <td>1</td> <td>0</td> <td>0</td> <td>1</td> <td>0DC00H</td> <td>8k/16k</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>0</td> <td>0C000H</td> <td>32k/32k</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> <td>1</td> <td>0C800H</td> <td>32k/32k</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>0</td> <td>0D000H</td> <td>32k/32k</td> </tr> <tr> <td>1</td> <td>1</td> <td>0</td> <td>1</td> <td>0D800H</td> <td>32k/32k</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>0</td> <td>0C000H</td> <td>64k/64k</td> </tr> <tr> <td>1</td> <td>1</td> <td>1</td> <td>1</td> <td>0D000H</td> <td>64k/64k</td> </tr> </tbody> </table>	Bit 3	Bit 2	Bit 1	Bit 0	Address	Size (I/O / Shared Mem.)	0	0	0	X	X	No boot PROM	0	0	1	0	0C000H	8k/16k	0	0	1	1	0C400H	8k/16k	0	1	0	0	0C800H	8k/16k	0	1	0	1	0CC00H	8k/16k	0	1	1	0	0D000H	8k/16k	0	1	1	1	0D400H	8k/16k	1	0	0	0	0D800H	8k/16k	1	0	0	1	0DC00H	8k/16k	1	0	1	0	0C000H	32k/32k	1	0	1	1	0C800H	32k/32k	1	1	0	0	0D000H	32k/32k	1	1	0	1	0D800H	32k/32k	1	1	1	0	0C000H	64k/64k	1	1	1	1	0D000H	64k/64k
Bit 3	Bit 2	Bit 1	Bit 0	Address	Size (I/O / Shared Mem.)																																																																																													
0	0	0	X	X	No boot PROM																																																																																													
0	0	1	0	0C000H	8k/16k																																																																																													
0	0	1	1	0C400H	8k/16k																																																																																													
0	1	0	0	0C800H	8k/16k																																																																																													
0	1	0	1	0CC00H	8k/16k																																																																																													
0	1	1	0	0D000H	8k/16k																																																																																													
0	1	1	1	0D400H	8k/16k																																																																																													
1	0	0	0	0D800H	8k/16k																																																																																													
1	0	0	1	0DC00H	8k/16k																																																																																													
1	0	1	0	0C000H	32k/32k																																																																																													
1	0	1	1	0C800H	32k/32k																																																																																													
1	1	0	0	0D000H	32k/32k																																																																																													
1	1	0	1	0D800H	32k/32k																																																																																													
1	1	1	0	0C000H	64k/64k																																																																																													
1	1	1	1	0D000H	64k/64k																																																																																													
4	COMP	<p><b>COMPATIBLE:</b> This bit determines if the AT/LANTIC Controller's memory and I/O maps are compatible with the EtherCard PLUS and Novell boards or if they use the full 64k address space available to the NIC. A low level indicates compatible mode.</p>																																																																																																
5	INTMOD	<p><b>INTERRUPT MODE:</b> When this bit is low the AT/LANTIC Controller is in Direct Drive interrupt mode. When it is high Coded Output interrupt mode is used.</p>																																																																																																
6	CLKSEL	<p><b>CLOCK SELECT:</b> If this bit is low the NIC core is clocked by the 20 MHz. If this bit is high the NIC core is clocked by the signal on the BSCLK pin.</p>																																																																																																
7	SOFEN	<p><b>SOFTWARE ENABLE:</b> If this bit is set low then the user can program configuration registers A and B in software. If this bit is set high then the configuration registers are not accessible. If EECONFIG is high, the configuration from the switches will be overwritten by the configuration from the EEPROM even if this bit is pulled high.</p>																																																																																																

## 5.0 Register Descriptions (Continued)

### 5.2 SHARED MEMORY MODE CONTROL REGISTERS

The following tables describe the functionality of the two control registers and the 8/16 detection registers.

#### Shared Memory AT Detect Register (Read only)

7	6	5	4	3	2	1	0
x	x	x	x	x	x	x	ATDET

Bits	Symbols	Function
D0	ATDET	<b>ATDET:</b> This bit shows the value on the DWID pin and can be read by software to determine whether the AT/LANTIC Controller is operating in an 8- or 16-bit slot. When this bit is read as a 1 the AT/LANTIC Controller is in a 16-bit slot (PC-AT system bus) and when read as a 0 it is in an 8-bit slot.

#### Shared Memory Control Register 1

7	6	5	4	3	2	1	0
RESET	MEME	A18	A17	A16	A15	A14	A13

Bits	Symbols	Function
D0-D5	A13-A18	<b>A13-18:</b> Lower part of the address register used to determine the position of the AT/LANTIC Controller's memory within the system memory map.
D6	MEME	<b>MEMORY ENABLE:</b> Enables external memory accesses when held high. This bit will power up low, so the user must program the base memory address and set this bit high to enable the memory into the system's memory map.
D7	RESET	<b>RESET:</b> Resets NIC core of AT/LANTIC Controller.

#### Shared Memory Control Register 2

7	6	5	4	3	2	1	0
8/16	MEMW	Unused	LA23	LA22	LA21	LA20	LA19

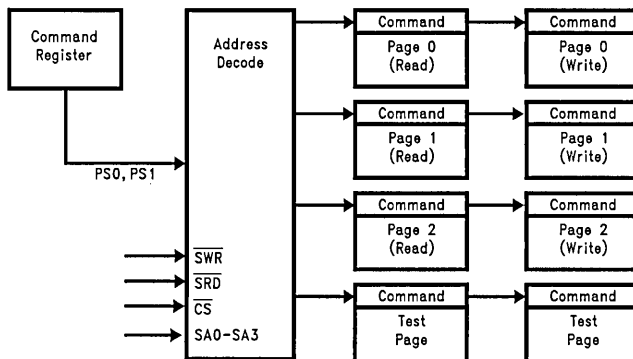
Bits	Symbols	Function
D0-D4	LA19-LA23	<b>LA19-23:</b> Upper part of the address register used to determine the position of the AT/LANTIC Controller's memory within the system memory map.
D5		<b>UNUSED</b>
D6	MEMW	<b>MEMORY WIDTH:</b> Sets width of external memory. When set low external memory is accessed as byte wide, so only 8 kbytes of memory are available. When set high external memory is accessed as word wide, so 16 kbytes are available. In non-compatible mode up to 64 kbytes of external memory is allowed when this bit is set high, or 32 kbytes when low. When bit 7 is set high this bit must also be set high.
D7	8/16	<b>8/16-BIT:</b> Allows 16-bit system accesses to external memory when set high. When low only 8-bit accesses are allowed. When high the generation of the M16 output is allowed.



## 5.0 Register Descriptions (Continued)

### 5.3 NIC CORE REGISTERS

All registers are 8-bit wide and mapped into two pages which are selected in the Command Register (PS0, PS1). Pins SA0–SA3 are used to address registers within each page. Page 0 registers are those registers which are commonly accessed during AT/LANTIC Controller operation while page 1 registers are used primarily for initialization. The registers are partitioned to avoid having to perform two write/read cycles to access commonly used registers.



TL/F/11498-20

FIGURE 23. NIC Core Register Mapping

## 5.0 Register Descriptions (Continued)

### Register Assignments

#### Page 0 Address Assignments (PS1 = 0, PS0 = 0)

SA0-SA3	RD	WR
00H	Command (CR)	Command (CR)
01H	Current Local DMA Address 0 (CLDA0)	Page Start Register (PSTART)
02H	Current Local DMA Address 1 (CLDA1)	Page Stop Register (PSTOP)
03H	Boundary Pointer (BNRY)	Boundary Pointer (BNRY)
04H	Transmit Status Register (TSR)	Transmit Page Start Address (TPSR)
05H	Number of Collisions Register (NCR)	Transmit Byte Count Register 0 (TBCR0)
06H	FIFO (FIFO)	Transmit Byte Count Register 1 (TBCR1)
07H	Interrupt Status Register (ISR)	Interrupt Status Register (ISR)
08H	Current Remote DMA Address 0 (CRDA0)	Remote Start Address Register 0 (RSAR0)
09H	Current Remote DMA Address 1 (CRDA1)	Remote Start Address Register 1 (RSAR1)
0AH	Reserved	Remote Byte Count Register 0 (RBCR0)
0BH	Reserved	Remote Byte Count Register 1 (RBCR1)
0CH	Receive Status Register (RSR)	Receive Configuration Register (RCR)
0DH	Tally Counter 0 (Frame Alignment Errors) (CNTR0)	Transmit Configuration Register (TCR)
0EH	Tally Counter 1 (CRC Errors) (CNTR1)	Data Configuration Register (DCR)
0FH	Tally Counter 2 Missed Packet Errors) (CNTR2)	Interrupt Mask Register (IMR)

## 5.0 Register Descriptions (Continued)

Page 1 Address Assignments (PS1 = 0, PS0 = 1)

SA0-SA3	RD	WR
00H	Command (CR)	Command (CR)
01H	Physical Address Register 0 (PAR0)	Physical Address Register 0 (PA R0)
02H	Physical Address Register 1 (PAR1)	Physical Address Register 1 (PA R1)
03H	Physical Address Register 2 (PAR2)	Physical Address Register 2 (PAR2)
04H	Physical Address Register 3 (PAR3)	Physical Address Register 3 (PAR3)
05H	Physical Address Register 4 (PAR4)	Physical Address Register 4 (PAR4)
06H	Physical Address Register 5 (PAR5)	Physical Address Register 5 (PAR5)
07H	Current Page Register (CURR)	Current Page Register (CURR)
08H	Multicast Address Register 0 (MAR0)	Multicast Address Register 0 (MAR0)
09H	Multicast Address Register 1 (MAR1)	Multicast Address Register 1 (MAR1)
0AH	Multicast Address Register 2 (MAR2)	Multicast Address Register 2 (MAR2)
0BH	Multicast Address Register 3 (MAR3)	Multicast Address Register 3 (MAR3)
0CH	Multicast Address Register 4 (MAR4)	Multicast Address Register 4 (MAR4)
0DH	Multicast Address Register 5 (MAR5)	Multicast Address Register 5 (MAR5)
0EH	Multicast Address Register 6 (MAR6)	Multicast Address Register 6 (MAR6)
0FH	Multicast Address Register 7 (MAR7)	Multicast Address Register 7 (MAR7)

## 5.0 Register Descriptions (Continued)

Page 2 Address Assignments (PS1 = 1, PS0 = 0)

SA0-SA3	RD	WR
00H	Command (CR)	Command (CR)
01H	Page Start Register (PSTART)	Current Local DMA Address 0 (CLDA0)
02H	Page Stop Register (PSTOP)	Current Local DMA Address 1 (CLDA1)
03H	Remote Next Packet Pointer	Remote Next Packet Pointer
04H	Transmit Page Start Address (TPSR)	Reserved
05H	Local Next Packet Pointer	Local Next Packet Pointer
06H	Address Counter (Upper)	Address Counter (Upper)
07H	Address Counter (Lower)	Address Counter (Lower)
08H	Reserved	Reserved
09H	Reserved	Reserved
0AH	Reserved	Reserved
0BH	Reserved	Reserved
0CH	Receive Configuration Register (RCR)	Reserved
0DH	Transmit Configuration Register (TCR)	Reserved
0EH	Data Configuration Register (DCR)	Reserved
0FH	Interrupt Mask Register (IMR)	Reserved

**Note:** Page 2 registers should only be accessed for diagnostic purposes. They should not be modified during normal operation.

Page 3 should never be modified.

## 5.0 Register Descriptions (Continued)

### COMMAND REGISTER (CR) 00H (READ/WRITE)

The Command Register is used to initiate transmissions, enable or disable Remote DMA operations and to select register pages. To issue a command the microprocessor sets the corresponding bit(s) (RD2, RD1, RD0, TXP). Further commands may be overlapped, but with the following rules: (1) If a transmit command overlaps with a remote DMA operation, bits RD0, RD1, and RD2 must be maintained for the remote DMA command when setting the TXP bit. Note, if a remote DMA command is re-issued when giving the transmit command, the DMA will complete immediately if the remote byte count register has not been re-initialized. (2) If a remote DMA operation overlaps a transmission, RD0, RD1, and RD2 may be written with the desired values and a "0" written to the TXP bit. Writing a "0" to this bit has no effect. (3) A remote write DMA may not overlap remote read operation or visa versa. Either of these operations must either complete or be aborted before the other operation may start. Bits PS1, PS0, RD2, and STP may be set any time.

7	6	5	4	3	2	1	0
PS1	PS0	RD2	RD1	RD0	TXP	STA	STP

Bits	Symbols	Description																								
D0	STP	<b>STOP:</b> Software reset command, takes the controller offline, no packets will be received or transmitted. Any reception or transmission in progress will continue to completion before entering the reset state. To exit this state, the STP bit must be reset and the STA bit must be set high. To perform a software reset, this bit should be set high. The software reset has executed only when indicated by the RST bit in the ISR being set to at 1. STP powers up high. <b>Note:</b> If the AT/LANTIC Controller has previously been in start mode and the STP is set, both the STP and STA bits will remain set.																								
D1	STA	<b>START:</b> This bit is used to activate the NIC Core after either power up, or when the NIC Core has been placed in a reset mode by software command or error. STA powers up low.																								
D2	TXP	<b>TRANSMIT PACKET:</b> This bit must be set to initiate transmission of a packet. TXP is internally reset either after the transmission is completed or aborted. This bit should be set only after the Transmit Byte Count and Transmit Page Start registers have been programmed.																								
D3–D5	RD0–RD2	<b>REMOTE DMA COMMAND:</b> These three encoded bits control operation of the Remote DMA channel. RD2 can be set to abort any Remote DMA command in progress. The Remote Byte Count Registers should be cleared when a Remote DMA has been aborted. The Remote Start Addresses are not restored to the starting address if the Remote DMA is aborted. <table border="1"> <thead> <tr> <th>RD2</th> <th>RD1</th> <th>RD0</th> <th></th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> <td>Not Allowed</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>Remote Read</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>Remote Write</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>Send Packet</td> </tr> <tr> <td>1</td> <td>X</td> <td>X</td> <td>Abort/Complete Remote DMA (Note 1)</td> </tr> </tbody> </table>	RD2	RD1	RD0		0	0	0	Not Allowed	0	0	1	Remote Read	0	1	0	Remote Write	0	1	1	Send Packet	1	X	X	Abort/Complete Remote DMA (Note 1)
RD2	RD1	RD0																								
0	0	0	Not Allowed																							
0	0	1	Remote Read																							
0	1	0	Remote Write																							
0	1	1	Send Packet																							
1	X	X	Abort/Complete Remote DMA (Note 1)																							
D6, D7	PS0, PS1	<b>PAGE SELECT:</b> These two encoded bits select which register page is to be accessed with addresses RA0–3. <table border="1"> <thead> <tr> <th>PS1</th> <th>PS0</th> <th></th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Register Page 0</td> </tr> <tr> <td>0</td> <td>1</td> <td>Register Page 1</td> </tr> <tr> <td>1</td> <td>0</td> <td>Register Page 2</td> </tr> <tr> <td>1</td> <td>1</td> <td>Reserved</td> </tr> </tbody> </table>	PS1	PS0		0	0	Register Page 0	0	1	Register Page 1	1	0	Register Page 2	1	1	Reserved									
PS1	PS0																									
0	0	Register Page 0																								
0	1	Register Page 1																								
1	0	Register Page 2																								
1	1	Reserved																								

**Note 1:** If a remote DMA operation is aborted and the remote byte count has not decremented to zero, the data transfer port should be read, for a remote read or send packet, or written to, for a remote write. This is required to ensure future correct operation.

## 5.0 Register Descriptions (Continued)

### INTERRUPT STATUS REGISTER (ISR) 07H (READ/WRITE)

This register is accessed by the host processor to determine the cause of an interrupt. Any interrupt can be masked in the Interrupt Mask Register (IMR). Individual interrupt bits are cleared by writing a "1" into the corresponding bit of the ISR. The valid interrupt output is active as long as any unmasked signal is set, and will not go low until all unmasked bits in this register have been cleared. The ISR must be cleared after power up by writing it with all 1's.

7	6	5	4	3	2	1	0
RST	RDC	CNT	OVW	TXE	RXE	PTX	PRX

Bits	Symbols	Description
D0	PRX	<b>PACKET RECEIVED:</b> Indicates packet received with no errors.
D1	PTX	<b>PACKET TRANSMITTED:</b> Indicates packet transmitted with no errors.
D2	RXE	<b>RECEIVE ERROR:</b> Indicates that a packet was received with one or more of the following errors: —CRC Error —Frame Alignment Error —FIFO Overrun —Missed Packet
D3	TXE	<b>TRANSMIT ERROR:</b> Set when packet transmitted with one or more of the following errors: —Excessive Collisions —FIFO Underrun
D4	OVW	<b>OVERWRITE WARNING:</b> Set when receive buffer ring storage resources have been exhausted. (Local DMA has reached Boundary Pointer)
D5	CNT	<b>COUNTER OVERFLOW:</b> Set when MSB of one or more of the Network Tally Counters has been set.
D6	RDC	<b>REMOTE DMA COMPLETE:</b> Set when Remote DMA operation has been completed.
D7	RST	<b>RESET STATUS:</b> Set when AT/LANTIC Controller enters reset state and cleared when a Start Command is issued to the CR. This bit is also set when a Receive Buffer Ring overflow occurs and is cleared when one or more packets have been removed from the ring. Writing to this bit has no effect. <b>Note:</b> This bit does not generate an interrupt, it is merely a status indicator.

## 5.0 Register Descriptions (Continued)

### INTERRUPT MASK REGISTER (IMR) 0FH (WRITE)

The Interrupt Mask Register is used to mask interrupts. Each interrupt mask bit corresponds to a bit in the Interrupt Status Register (ISR). If an interrupt mask bit is set, an interrupt will be issued whenever the corresponding bit in the ISR is set. If any bit in the IMR is set low, an interrupt will not occur when the bit in the ISR is set. **The IMR powers up all zeros.**

7	6	5	4	3	2	1	0
—	RDCE	CNTE	OVWE	TXEE	RXEE	PTXE	PRXE

Bits	Symbols	Description
D0	PRXE	<b>PACKET RECEIVED INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when packet received
D1	PTXE	<b>PACKET TRANSMITTED INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when packet is transmitted
D2	RXEE	<b>RECEIVE ERROR INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when packet received with error
D3	TXEE	<b>TRANSMIT ERROR INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when packet transmission results in error
D4	OVWE	<b>OVERWRITE WARNING INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when Buffer Management Logic lacks sufficient buffers to store incoming packet
D5	CNTE	<b>COUNTER OVERFLOW INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when MSB of one or more of the Network Statistics counters has been set
D6	RDCE	<b>DMA COMPLETE INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when Remote DMA transfer has been completed
D7	reserved	reserved

## 5.0 Register Descriptions (Continued)

### DATA CONFIGURATION REGISTER (DCR) 0EH (WRITE)

This Register is used to program the AT/LANTIC Controller for 8- or 16-bit memory interface, select byte ordering in 16-bit applications and establish FIFO thresholds. **The DCR must be initialized prior to loading the Remote Byte Count Registers. LAS is set on power up.**

7	6	5	4	3	2	1	0
—	FT1	FT0	ARM	LS	LAS	BOS	WTS

Bits	Symbols	Description																								
D0	WTS	<p><b>WORD TRANSFER SELECT</b>                      0: Selects byte-wide DMA transfers                      1: Selects word-wide DMA transfers                      ;WTS establishes byte or word transfers for both Remote and Local DMA transfers  <b>Note:</b> When word-wide mode is selected, up to 32k words are addressable; A0 remains low.</p>																								
D1	BOS	<p><b>BYTE ORDER SELECT</b>                      0: MS byte placed on AD15–AD8 and LS byte on AD7–AD0. (32xxx, 80x86)                      1: MS byte placed on AD7–AD0 and LS byte on AD15–AD8. (680x0)                      ;Ignored when WTS is low</p>																								
D2	LAS	<p><b>LONG ADDRESS SELECT</b>                      0: Dual 16-bit DMA mode                      1: Single 32-bit DMA mode                        ;When LAS is high, the contents of the Remote DMA registers RSAR0, 1 are issued as A16–A31 Power up high</p>																								
D3	LS	<p><b>LOOPBACK SELECT</b>                      0: Loopback mode selected. Bits D1 and D2 of the TCR must also be programmed for Loopback operation                      1: Normal Operation</p>																								
D4	ARM	<p><b>AUTO-INITIALIZE REMOTE</b>                      0: Send Command not executed, all packets removed from Buffer Ring under program control                      1: Send Command executed, Remote DMA auto-initialized to remove packets from Buffer Ring  <b>Note:</b> Send Command cannot be used with 680x0 byte processors.</p>																								
D5 and D6	FT0 and FT1	<p><b>FIFO THRESHOLD SELECT:</b> Encoded FIFO threshold. Establishes point at which the memory bus is requested when filling or emptying the FIFO. During reception, the FIFO threshold indicates the number of bytes (or words) the FIFO has filled serially from the network before the FIFO is emptied onto the memory bus.  <b>Note:</b> FIFO threshold setting determines the DMA burst length.</p> <table border="1" style="margin-left: 20px;"> <thead> <tr> <th colspan="4">Receive Thresholds</th> </tr> <tr> <th>FT1</th> <th>FT0</th> <th>Word Wide</th> <th>Byte Wide</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>1 Word</td> <td>2 Bytes</td> </tr> <tr> <td>0</td> <td>1</td> <td>2 Words</td> <td>4 Bytes</td> </tr> <tr> <td>1</td> <td>0</td> <td>4 Words</td> <td>8 Bytes</td> </tr> <tr> <td>1</td> <td>1</td> <td>6 Words</td> <td>12 Bytes</td> </tr> </tbody> </table> <p>During transmission, the FIFO threshold indicates the number of bytes (or words) the FIFO has filled from the Local DMA before being transferred to the memory. Thus, the transmission threshold is 13 bytes less the received threshold.</p>	Receive Thresholds				FT1	FT0	Word Wide	Byte Wide	0	0	1 Word	2 Bytes	0	1	2 Words	4 Bytes	1	0	4 Words	8 Bytes	1	1	6 Words	12 Bytes
Receive Thresholds																										
FT1	FT0	Word Wide	Byte Wide																							
0	0	1 Word	2 Bytes																							
0	1	2 Words	4 Bytes																							
1	0	4 Words	8 Bytes																							
1	1	6 Words	12 Bytes																							



## 5.0 Register Descriptions (Continued)

### TRANSMIT CONFIGURATION REGISTER (TCR) ODH (WRITE)

The transmit configuration establishes the actions of the transmitter section of the AT/LANTIC Controller during transmission of a packet on the network. **LB1 and LB0 which select loopback mode power up as 0.**

7	6	5	4	3	2	1	0
—	—	—	OFST	ATD	LB1	LB0	CRC

Bits	Symbols	Description																				
D0	CRC	<b>INHIBIT CRC</b> 0: CRC appended by transmitter 1: CRC inhibited by transmitter In loopback mode CRC can be enabled or disabled to test the CRC logic																				
D1 and D2	LB0 and LB1	<b>ENCODED LOOPBACK CONTROL:</b> These encoded configuration bits set the type of loopback that is to be performed. Note that loopback in mode 2 places the ENDEC Module in loopback mode and that D3 of the DCR must be set to zero for loopback operation. <table border="1" style="margin-left: 20px;"> <thead> <tr> <th></th> <th>LB1</th> <th>LB0</th> <th></th> </tr> </thead> <tbody> <tr> <td>Mode 0</td> <td>0</td> <td>0</td> <td>Normal Operation (LPBK = 0)</td> </tr> <tr> <td>Mode 1</td> <td>0</td> <td>1</td> <td>Internal NIC Module Loopback (LPBK = 0)</td> </tr> <tr> <td>Mode 2</td> <td>1</td> <td>0</td> <td>Internal ENDEC Module Loopback (LPBK = 1)</td> </tr> <tr> <td>Mode 3</td> <td>1</td> <td>1</td> <td>External Loopback (LPBK = 0)</td> </tr> </tbody> </table>		LB1	LB0		Mode 0	0	0	Normal Operation (LPBK = 0)	Mode 1	0	1	Internal NIC Module Loopback (LPBK = 0)	Mode 2	1	0	Internal ENDEC Module Loopback (LPBK = 1)	Mode 3	1	1	External Loopback (LPBK = 0)
	LB1	LB0																				
Mode 0	0	0	Normal Operation (LPBK = 0)																			
Mode 1	0	1	Internal NIC Module Loopback (LPBK = 0)																			
Mode 2	1	0	Internal ENDEC Module Loopback (LPBK = 1)																			
Mode 3	1	1	External Loopback (LPBK = 0)																			
D3	ATD	<b>AUTO TRANSMIT DISABLE:</b> This bit allows another station to disable the AT/LANTIC Controller's transmitter by transmission of a particular multicast packet. The transmitter can be re-enabled by resetting this bit or by reception of a second particular multicast packet. 0: Normal Operation 1: Reception of multicast address hashing to bit 62 disables transmitter, reception of multicast address hashing to bit 63 enables transmitter.																				
D4	OFST	<b>COLLISION OFFSET ENABLE:</b> This bit modifies the backoff algorithm to allow prioritization of nodes. 0: Backoff Logic implements normal algorithm. 1: Forces Backoff algorithm modification to 0 to $2^{\min(3 + n, 10)}$ slot times for first three collisions, then follows standard backoff. (For the first three collisions, the station has higher average backoff delay making a low priority mode.)																				
D5	reserved	reserved																				
D6	reserved	reserved																				
D7	reserved	reserved																				

## 5.0 Register Descriptions (Continued)

### TRANSMIT STATUS REGISTER (TSR) 04H (READ)

This register records events that occur on the media during transmission of a packet. It is cleared when the next transmission is initiated by the host. All bits remain low unless the event that corresponds to a particular bit occurs during transmission. Each transmission should be followed by a read of this register. The contents of this register are not specified until after the first transmission.

7	6	5	4	3	2	1	0
OWC	CDH	FU	CRS	ABT	COL	—	PTX

Bits	Symbols	Description
D0	PTX	<b>PACKET TRANSMITTED:</b> Indicates transmission without error. (No excessive collisions or FIFO underrun)(ABT = "0", FU = "0")
D1	reserved	reserved
D2	COL	<b>TRANSMIT COLLIDED:</b> Indicates that the transmission collided at least once with another station on the network. The number of collisions is recorded in the Number of Collisions Registers (NCR).
D3	ABT	<b>TRANSMIT ABORTED:</b> Indicates the AT/LANTIC Controller aborted transmission because of excessive collisions. (Total number of transmissions including original transmission attempt equals 16.)
D4	CRS	<b>CARRIER SENSE LOST:</b> This bit is set when carrier is lost during transmission of the packet. Transmission is not aborted on loss of carrier.
D5	FU	<b>FIFO UNDERRUN:</b> If the AT/LANTIC Controller cannot gain access of the bus before the FIFO empties, this bit is set. Transmission of the packet will be aborted.
D6	CDH	<b>CD HEARTBEAT:</b> Failure of the transceiver to transmit a collision signal after transmission of a packet will set this bit. The Collision Detect (CD) heartbeat signal must commence during the first 6.4 $\mu$ s of the Interframe Gap following a transmission. In certain collisions, the CD Heartbeat bit will be set even though the transceiver is not performing the CD heartbeat test.
D7	OWC	<b>OUT OF WINDOW COLLISION:</b> Indicates that a collision occurred after a slot time (51.2 $\mu$ s). Transmissions rescheduled as in normal collisions.

## 5.0 Register Descriptions (Continued)

### RECEIVE CONFIGURATION REGISTER (RCR) 0CH (WRITE)

This register determines operation of the AT/LANTIC Controller during reception of a packet and is used to program what types of packets to accept.

7	6	5	4	3	2	1	0
—	—	MON	PRO	AM	AB	AR	SEP

Bits	Symbols	Description
D0	SEP	<b>SAVE ERRORED PACKETS</b> 0: Packets with receive errors are rejected. 1: Packets with receive errors are accepted. Receive errors are CRC and Frame Alignment errors.
D1	AR	<b>ACCEPT RUNT PACKETS:</b> This bit allows the receiver to accept packets that are smaller than 64 bytes. The packet must be at least 8 bytes long to be accepted as a runt. 0: Packets with fewer than 64 bytes rejected. 1: Packets with fewer than 64 bytes accepted.
D2	AB	<b>ACCEPT BROADCAST:</b> Enables the receiver to accept a packet with an all 1's destination address. 0: Packets with broadcast destination address rejected. 1: Packets with broadcast destination address accepted.
D3	AM	<b>ACCEPT MULTICAST:</b> Enables the receiver to accept a packet with a multicast address, all multicast addresses must pass the hashing array. 0: Packets with multicast destination address not checked. 1: Packets with multicast destination address checked.
D4	PRO	<b>PROMISCUOUS PHYSICAL:</b> Enables the receiver to accept all packets with a physical address. 0: Physical address of node must match the station address programmed in PAR0–PAR5. 1: All packets with physical addresses accepted.
D5	reserved	reserved (program to 0)
D6	reserved	reserved
D7	reserved	reserved

**Note:** D2 and D3 are "OR'd" together, i.e., if D2 and D3 are set the AT/LANTIC Controller will accept broadcast and multicast addresses as well as its own physical address. To establish full promiscuous mode, bits D2, D3, and D4 should be set. In addition the multicast hashing array must be set to all 1's in order to accept all multicast addresses.

## 5.0 Register Descriptions (Continued)

### RECEIVE STATUS REGISTER (RSR) OCH (READ)

This register records status of the received packet, including information on errors and the type of address match, either physical or multicast. The contents of this register are written to buffer memory by the DMA after reception of a good packet. If packets with errors are to be saved the receive status is written to memory at the head of the erroneous packet if an erroneous packet is received. If packets with errors are to be rejected the RSR will not be written to memory. The contents will be cleared when the next packet arrives. CRC errors, Frame Alignment errors and missed packets are counted internally by the AT/LAN-TIC Controller which relinquishes the Host from reading the RSR in real time to record errors for Network Management Functions. The contents of this register are not specified until after the first reception.

7	6	5	4	3	2	1	0
DFR	DIS	PHY	MPA	FO	FAE	CRC	PRX

Bits	Symbols	Description
D0	PRX	<b>PACKET RECEIVED INTACT:</b> Indicates packet received without error. (Bits CRC, FAE, FO, and MPA are zero for the received packet.)
D1	CRC	<b>CRC ERROR:</b> Indicates packet received with CRC error. Increments Tally Counter (CNTR1). This bit will also be set for Frame Alignment errors.
D2	FAE	<b>FRAME ALIGNMENT ERROR:</b> Indicates that the incoming packet did not end on a byte boundary and the CRC did not match at last byte boundary. Increments Tally Counter (CNTR0).
D3	FO	<b>FIFO OVERRUN:</b> This bit is set when the FIFO is not serviced causing overflow during reception. Reception of the packet will be aborted.
D4	MPA	<b>MISSED PACKET:</b> Set when packet intended for node cannot be accepted by SNIC because of a lack of receive buffers or if the controller is in monitor mode and did not buffer the packet to memory. Increments Tally Counter (CNTR2).
D5	PHY	<b>PHYSICAL/MULTICAST ADDRESS:</b> Indicates whether received packet had a physical or multicast address type. 0: Physical Address Match 1: Multicast/Broadcast Address Match
D6	DIS	<b>RECEIVER DISABLED:</b> Set when receiver disabled by entering Monitor mode. Reset when receiver is re-enabled when exiting Monitor mode.
D7	DFR	<b>DEFERRING:</b> Set when internal Carrier Sense or Collision signals are generated in the ENDEC module. If the transceiver has asserted the CD line as a result of the jabber, this bit will stay set indicating the jabber condition.

**Note:** Following coding applies to CRC and FAE bits

FAE	CRC	Type of Error
0	0	No Error (Good CRC and <6 Dribble Bits)
0	1	CRC Error
1	0	Illegal, will not occur
1	1	Frame Alignment Error and CRC Error

## 5.0 Register Descriptions (Continued)

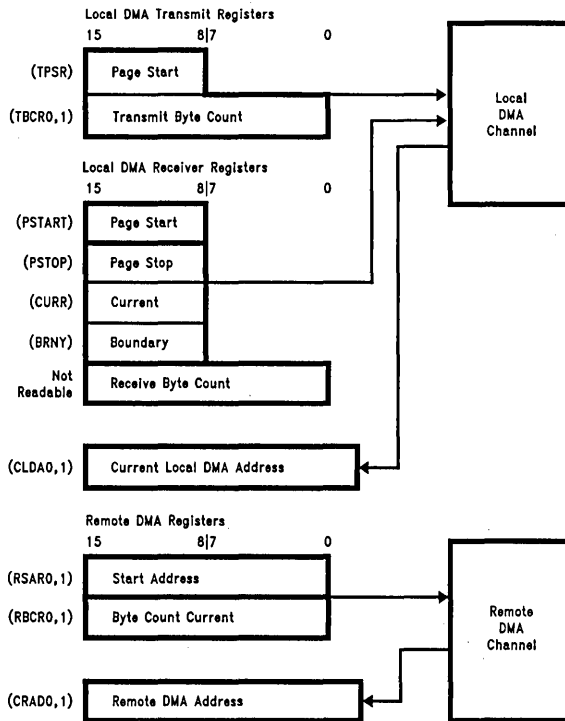


FIGURE 24. DMA Register

TL/F11498-21

**Note:** In the figure above, registers are shown as 8- or 16-bits wide. Although some registers are 16-bit internal registers, all registers are accessed as 8-bit registers. Thus the 16-bit Transmit Byte Count Register is broken into two 8-bit registers, TBCR0 and TBCR1. Also TPSR, PSTART, PSTOP, CURR and BRNY only check or control the upper 8 bits of address information on the bus. Thus they are shifted to positions 15-8 in the diagram above.

### 5.4 DP8390 Core DMA Registers

The DMA Registers are partitioned into groups; Transmit, Receive and Remote DMA Registers. The Transmit registers are used to initialize the Local DMA Channel for transmission of packets while the Receive Registers are used to initialize the Local DMA Channel for packet Reception. The Page Stop, Page Start, Current and Boundary Registers are used by the Buffer Management Logic to supervise the Receive Buffer Ring. The Remote DMA Registers are used to initialize the Remote DMA.

#### Transmit DMA Registers

##### TRANSMIT PAGE START REGISTER (TPSR)

This register points to the assembled packet to be transmitted. Only the eight higher order addresses are specified since all transmit packets are assembled on 256-byte page boundaries. The bit assignment is shown below. The values placed in bits D7-D0 will be used to initialize the higher order address (A8-A15) of the Local DMA for transmission. The lower order bits (A7-A0) are initialized to zero.

#### Bit Assignment

	7	6	5	4	3	2	1	0
TPSR	A15	A14	A13	A12	A11	A10	A9	A8

(A7-A0 Initialized to 0)

##### TRANSMIT BYTE COUNT REGISTER 0,1 (TBCR0, TBCR1)

These two registers indicate the length of the packet to be transmitted in bytes. The count must include the number of bytes in the source, destination, length and data fields. The maximum number of transmit bytes allowed is 64 kbytes. The AT/LANTIC Controller will not truncate transmissions longer than 1500 bytes. The bit assignment is shown below:

	7	6	5	4	3	2	1	0
TBCR1	L15	L14	L13	L12	L11	L10	L9	L8

	7	6	5	4	3	2	1	0
TBCR0	L7	L6	L5	L4	L3	L2	L1	L0

#### Local DMA Receive Registers

##### PAGE START STOP REGISTERS (PSTART, PSTOP)

The Page Start and Page Stop Registers program the starting and stopping address of the Receive Buffer Ring. Since

## 5.0 Register Descriptions (Continued)

the AT/LANTIC Controller uses fixed 256-byte buffers aligned on page boundaries only the upper eight bits of the start and stop address are specified.

PSTART, PSTOP bit assignment

	7	6	5	4	3	2	1	0
PSTART	A15	A14	A13	A12	A11	A10	A9	A8
PSTOP	A15	A14	A13	A12	A11	A10	A9	A8

### BOUNDARY (BNRY) REGISTER

This register is used to prevent overflow of the Receive Buffer Ring. Buffer management compares the contents of this register to the next buffer address when linking buffers together. If the contents of this register match the next buffer address the Local DMA operation is aborted.

	7	6	5	4	3	2	1	0
BNRY	A15	A14	A13	A12	A11	A10	A9	A8

### CURRENT PAGE REGISTER (CURR)

This register is used internally by the Buffer Management Logic as a backup register for reception. CURR contains the address of the first buffer to be used for a packet reception and is used to restore DMA pointers in the event of receive errors. This register is initialized to the same value as PSTART and should not be written to again unless the controller is Reset.

	7	6	5	4	3	2	1	0
CURR	A15	A14	A13	A12	A11	A10	A9	A8

### CURRENT LOCAL DMA REGISTER 0,1 (CLDA0,1)

These two registers can be accessed to determine the current Local DMA Address.

	7	6	5	4	3	2	1	0
CLDA1	A15	A14	A13	A12	A11	A10	A9	A8
	7	6	5	4	3	2	1	0
CLDA0	A7	A6	A5	A4	A3	A2	A1	A0

### Remote DMA Registers

#### REMOTE START ADDRESS REGISTERS (RSAR0,1)

Remote DMA operations are programmed via the Remote Start Address (RSAR0,1) and Remote Byte Count (RBCR0,1) registers. The Remote Start Address is used to point to the start of the block of data to be transferred and the Remote Byte Count is used to indicate the length of the block (in bytes).

	7	6	5	4	3	2	1	0
RSAR1	A15	A14	A13	A12	A11	A10	A9	A8
	7	6	5	4	3	2	1	0
RSAR0	A7	A6	A5	A4	A3	A2	A1	A0

#### REMOTE BYTE COUNT REGISTERS (RBCR0,1)

	7	6	5	4	3	2	1	0
RBCR1	A15	A14	A13	A12	A11	A10	A9	A8
	7	6	5	4	3	2	1	0
RBCR0	A7	A6	A5	A4	A3	A2	A1	A0

#### Notes:

RSAR0 programs the start address bits A0–A7.

RSAR1 programs the start address bits A8–A15.

Address incremented by two for word transfers, and by one for byte transfers. Byte count decremented by two for word transfers and by one for byte transfers.

RBCR0 programs LSB byte count.

RBCR1 programs MSB byte count.

#### CURRENT REMOTE DMA ADDRESS (CRDA0, CRDA1)

The Current Remote DMA Registers contain the current address of the Remote DMA. The bit assignment is shown below:

	7	6	5	4	3	2	1	0
CRDA1	A15	A14	A13	A12	A11	A10	A9	A8
	7	6	5	4	3	2	1	0
CRDA0	A7	A6	A5	A4	A3	A2	A1	A0

#### Physical Address Registers (PAR0–PAR5)

The physical address registers are used to compare the destination address of incoming packets for rejecting or accepting packets. Comparisons are performed on a byte-wide basis. The bit assignment shown below relates the sequence in PAR0–PAR5 to the bit sequence of the received packet.

	D7	D6	D5	D4	D3	D2	D1	D0
PAR0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
PAR1	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
PAR2	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
PAR3	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
PAR4	DA39	DA38	DA37	DA36	DA35	DA34	DA33	DA32
PAR5	DA47	DA46	DA45	DA44	DA43	DA42	DA41	DA40

	Destination Address					Source			
P/S	DA0	DA1	DA2	DA3	...	DA46	DA47	SA0	...

Note: P/S = Preamble, Sync  
DA0 = Physical/Multicast Bit

#### Multicast Address Registers (MAR0–MAR7)

The multicast address registers provide filtering of multicast addresses hashed by the CRC logic. All destination addresses are fed through the CRC logic and as the last bit of the destination address enters the CRC, the 6 most significant bits of the CRC generator are latched. These 6 bits are then decoded by a 1 of 64 decode to index a unique filter bit (FB0–63) in the multicast address registers. If the filter bit selected is set, the multicast packet is accepted. The system designer would use a program to determine which filter

## 5.0 Register Descriptions (Continued)

bits to set in the multicast registers. All multicast filter bits that correspond to multicast address accepted by the node are then set to one. To accept all multicast packets all of the registers are set to all ones.

**Note:** Although the hashing algorithm does not guarantee perfect filtering of multicast address, it will perfectly filter up to 64 multicast addresses if these addresses are chosen to map into unique locations in the multicast filter.

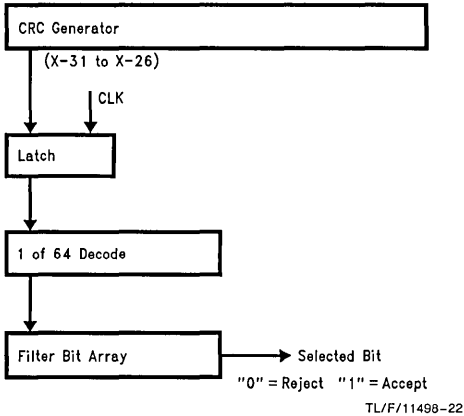


FIGURE 25. Multicast Addressing

## 6.0 Operation of AT/LANTIC Controller

This section details the operation of the AT/LANTIC Controller. The operations discussed are packet reception and transmission, bus operations, and loopback diagnostics.

### 6.1 TRANSMIT/RECEIVE PACKET ENCAPSULATION/DECAPSULATION

A standard IEEE 802.3 packet consists of the following fields: preamble, Start of Frame Delimiter (SFD), destination address, source address, length, data, and Frame Check Sequence (FCS). The typical format is shown in the figure following. The packets are Manchester encoded and decoded by the ENDEC module and transferred serially to the NIC module using NRZ data with a clock. All fields are of fixed length except for the data field. The AT/LANTIC Controller generates and appends the preamble, SFD and FCS field during transmission. The Preamble and SFD fields are stripped during reception. (The CRC is passed through to buffer memory during reception.)

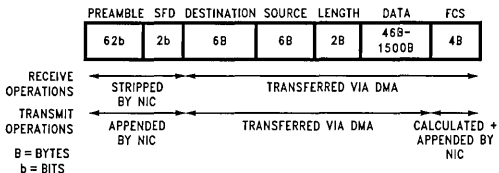


FIGURE 26. Ethernet Packet

### PREAMBLE AND START OF FRAME DELIMITER (SFD)

The Manchester encoded alternating 1,0 preamble field is used by the ENDEC to acquire bit synchronization with an incoming packet. When transmitted each packet contains 62 bits of alternating 1,0 preamble. Some of this preamble will be lost as the packet travels through the network. The preamble field is stripped by the NIC module. Byte alignment is performed with the Start of Frame Delimiter (SFD) pattern which consists of two consecutive 1's. The AT/LANTIC Controller does not treat the SFD pattern as a byte, it detects only the two-bit pattern. This allows any preceding preamble within the SFD to be used for phase locking.

### Destination Address

The destination address indicates the destination of the packet on the network and is used to filter unwanted packets from reaching a node. There are three types of address formats supported by the AT/LANTIC: physical, multicast and broadcast. The physical address is a unique address that corresponds only to a single node. All physical addresses have an MSB of "0". These addresses are compared to the internally stored physical address registers. Each bit in the destination address must match in order for the AT/LANTIC Controller to accept the packet. Multicast addresses begin with an MSB of "1". The AT/LANTIC Controller filters multicast addresses using a standard hashing algorithm that maps all multicast addresses into a 6-bit value. This 6-bit value indexes a 64-bit array that filters the value. If the address consists of all 1's it is a broadcast address, indicating that the packet is intended for all nodes. A promiscuous mode allows reception of all packets: the destination address is not required to match any filters. Physical, broadcast, multicast, and promiscuous address modes can be selected.

### Source Address

The source address is the physical address of the node that sent the packet. Source addresses cannot be multicast or broadcast addresses. This field is simply passed to buffer memory.

### Length Field

The 2-byte length field indicates the number of bytes that are contained in the data field of the packet. This field is not interpreted by the AT/LANTIC Controller.

### Data Field

The data field consists of anywhere from 46 to 1500 bytes. Messages longer than 1500 bytes need to be broken into multiple packets. Messages shorter than 46 bytes will require appending a pad to bring the data field to the minimum length of 46 bytes. If the data field is padded, the number of valid data bytes is indicated in the length field. **The AT/LANTIC Controller does not strip or append pad bytes for short packets, or check for oversize packets.**

### FCS Field

The Frame Check Sequence (FCS) is a 32-bit CRC field calculated and appended to a packet during transmission to allow detection of errors when a packet is received. During reception, error free packets result in a specific pattern in

## 6.0 Operation of AT/LANTIC Controller (Continued)

the CRC generator. Packets with improper CRC will be rejected. The AUTODIN II ( $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$ ) polynomial is used for the CRC calculations.

### 6.2 BUFFER MEMORY ACCESS CONTROL (DMA)

The buffer memory control capabilities of the AT/LANTIC Controller greatly simplify the use of the AT/LANTIC Controller in typical configurations. The local DMA channel transfers data between the FIFO and memory. On transmission, the packet is DMA'd from memory to the FIFO in bursts. Should a collision occur (up to 15 times), the packet is re-transmitted with no processor intervention. On reception, packets are moved via DMA from the FIFO to the receive buffer ring (as explained below).

A Remote DMA channel is also provided on the AT/LANTIC Controller to accomplish transfers between a buffer memory and an internal Data Port when using the AT/LANTIC Controller in I/O Mode. This Remote DMA channel is not used when the AT/LANTIC Controller is used in a shared Memory mode. In this second mode the buffer memory is dual ported, and directly mapped into the system memory. In this mode the system CPU directly accesses the RAM under software control to transfer packet data.

The following sections describe the operation of the Local DMA channel for packet reception which is used in both modes. For Shared Memory mode the description of the Remote DMA does not apply.

For reference an example configuration using the AT/LANTIC Controller is shown in *Figure 27*.

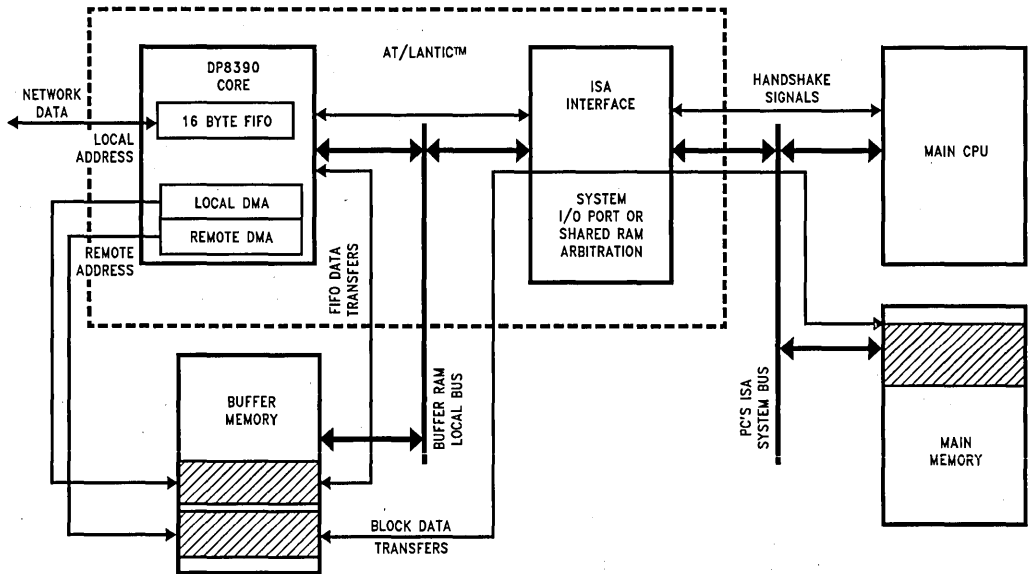


FIGURE 27. AT/LANTIC Controller Bus Architecture

TL/F/11498-24



## 6.0 Operation of AT/LANTIC Controller (Continued)

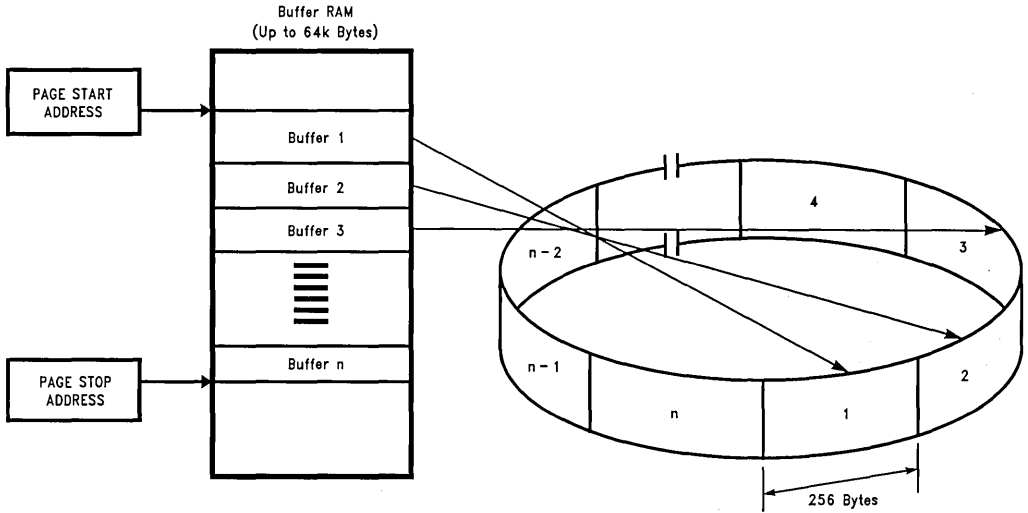


FIGURE 28. AT/LANTIC Controller Receiver Buffer Ring

### 6.3 PACKET RECEPTION

The Local DMA receive channel uses a Buffer Ring Structure comprised of a series of contiguous fixed length 256 byte (128 word) buffers for storage of received packets. The location of the Receive Buffer Ring is programmed in two registers, a Page Start and a Page Stop Register. Ethernet packets consist of a distribution of shorter link control packets and longer data packets, the 256 byte buffer length provides a good compromise between short packets and longer packets to most efficiently use memory. In addition these buffers provide memory resources for storage of back-to-back packets in loaded networks. The assignment of buffers for storing packets is controlled by Buffer Management Logic in the AT/LANTIC Controller. The Buffer Management Logic provides three basic functions: linking receive buffers for long packets, recovery of buffers when a packet is rejected, and recirculation of buffer pages that have been read by the host.

At initialization, a portion of the 64 kbyte (or 32 kword) address space is reserved for the receive buffer ring. Two eight bit registers, the Page Start Address Register (PSTART) and the Page Stop Address Register (PSTOP) define the physical boundaries of where the buffers reside. The AT/LANTIC Controller treats the list of buffers as a logical ring; whenever the DMA address reaches the Page Stop Address, the DMA is reset to the Page Start Address.

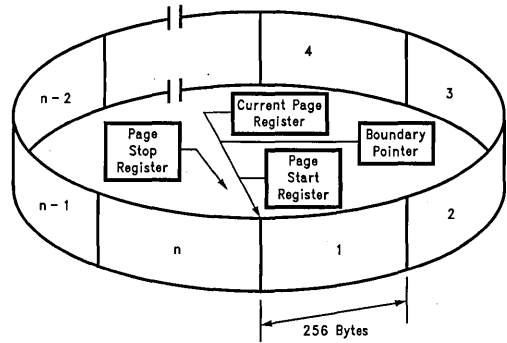


FIGURE 29. Buffer Ring at Initialization

#### Initialization of the Buffer Ring

Two static registers and two working registers control the operation of the Buffer Ring. These are the Page Start Register, Page Stop Register (both described previously), the Current Page Register and the Boundary Pointer Register. The Current Page Register points to the first buffer used to store a packet and is used to restore the DMA for writing status to the Buffer Ring or for restoring the DMA address in the event of a Runt packet, a CRC, or Frame Alignment error. The Boundary Register points to the first packet in the

1

## 6.0 Operation of AT/LANTIC Controller (Continued)

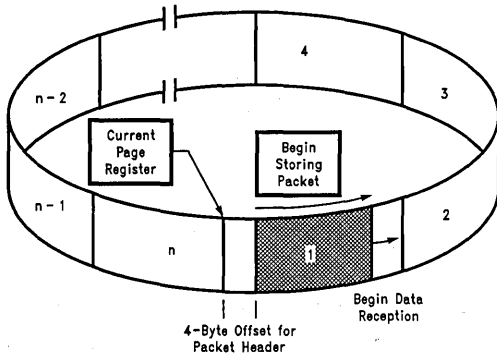
Ring not yet read by the host. If the local DMA address ever reaches the Boundary, reception is aborted. The Boundary Pointer is also used to initialize the Remote DMA for removing a packet and is advanced when a packet is removed. A simple analogy to remember the function of these registers is that the Current Page Register acts as a Write Pointer and the Boundary Pointer acts as a Read Pointer.

**Note 1:** At initialization, the Page Start Register value should be loaded into both the Current Page Register and the Boundary Pointer Register.

**Note 2:** The Page Start Register must not be initialized to 00H.

### Beginning of Reception

When the first packet begins arriving the AT/LANTIC Controller begins storing the packet at the location pointed to by the Current Page Register. An offset of 4 bytes is saved in this first buffer to allow room for storing receive status corresponding to this packet.



TL/F/11498-27

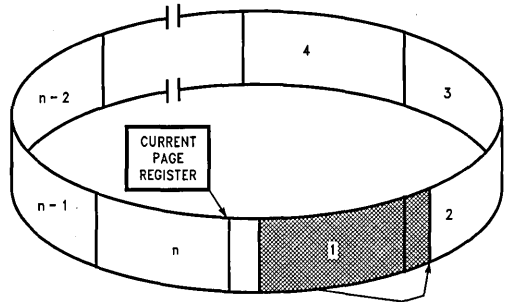
**FIGURE 30. Received Packet Enters the Buffer Pages**

### Linking Receive Buffer Pages

If the length of the packet exhausts the first 256 byte buffer, the DMA performs a forward link to the next buffer to store the remainder of the packet. For a maximal length packet the buffer logic will link six buffers to store the entire packet. Buffers cannot be skipped when linking, a packet will always be stored in contiguous buffers. Before the next buffer can be linked, the Buffer Management Logic performs two comparisons. The first comparison tests for equality between the DMA address of the next buffer and the contents of the Page Stop Register. If the buffer address equals the Page Stop Register, the buffer management logic will restore the DMA to the first buffer in the Receive Buffer Ring value programmed in the Page Start Address Register. The second comparison tests for equality between the DMA address of the next buffer address and the contents of the Boundary Pointer Register. If the two values are equal the reception is aborted. The Boundary Pointer Register can be used to protect against overwriting any area in the receive buffer ring that has not yet been read. When linking buffers, buffer management will never cross this pointer, effectively avoiding any overwrites. If the buffer address does not match either the Boundary Pointer or Page Stop Address, the link to the next buffer is performed.

### Linking Buffers

Before the DMA can enter the next contiguous 256 byte buffer, the address is checked for equality to PSTOP and to the Boundary Pointer. If neither are reached, the DMA is allowed to use the next buffer.



- 1) Check for = PSTOP
- 2) Check for = Boundary

TL/F/11498-28

**FIGURE 31. Linking Receive Buffer Pages**

### Buffer Ring Overflow

If the Buffer Ring has been filled and the DMA reaches the Boundary Pointer Address, reception of the incoming packet will be aborted by the AT/LANTIC Controller. Thus, the packets previously received and still contained in the Ring will not be destroyed.

In heavily loaded networks which cause overflows of the Receive Buffer Ring, the AT/LANTIC Controller may disable the local DMA and suspend further receptions even if the Boundary register is advanced beyond the Current register. In the event that the AT/LANTIC Controller should encounter a receive buffer overflow, it is necessary to implement the following routine. A receive buffer overflow is indicated by the AT/LANTIC Controller's assertion of the overflow bit (OVW) in the Interrupt Status Register (ISR).

If this routine is not adhered to, the AT/LANTIC Controller may act in an unpredictable manner. It should also be noted that it is not permissible to service an overflow interrupt by continuing to empty packets from the receive buffer without implementing the prescribed overflow routine. A flow chart of the AT/LANTIC™ Controller's overflow routine can be found in *Figure 32*.

**Note:** It is necessary to define a variable in the driver, which will be called "Resend".

1. Read and store the value of the TXP bit in the AT/LANTIC Controller's Command Register.
2. Issue the STOP command to the AT/LANTIC Controller. This is accomplished by setting the STP bit in the AT/LANTIC Controller's Command Register. Writing 21 H to the Command Register will stop the AT/LANTIC Controller.
3. Wait for at least 1.6 ms. Since the AT/LANTIC Controller will complete any transmission or reception that is in progress, it is necessary to time out for the maximum possible duration of an Ethernet transmission or reception. By waiting 1.6 ms this is achieved with some guard band added. Previously, it was recommended that the RST bit of the Interrupt Status Register be polled to insure that the pending transmission or reception is completed. This bit is not a reliable indicator and subsequently should be ignored.

## 6.0 Operation of AT/LANTIC Controller (Continued)

4. Clear the AT/LANTIC Controller's Remote Byte Count registers (RBCR0 and RBCR1).

5. Read the stored value of the TXP bit from step 1, above.

If this value is a 0, set the "Resend" variable to a 0 and jump to step 6.

If this value is a 1, read the AT/LANTIC Controller's Interrupt Status Register. If either the Packet Transmitted bit (PTX) or Transmit Error bit (TXE) is set to a 1, set the "Resend" variable to a 0 and jump to step 6. If neither of these bits is set, place a 1 in the "Resend" variable and jump to step 6.

This step determines if there was a transmission in progress when the stop command was issued in step 2. If there was a transmission in progress, the AT/LANTIC Controller's ISR is read to determine whether or not the packet was recognized by the AT/LANTIC Controller. If neither the PTX nor TXE bit was set, then the packet will essentially be lost and re-transmitted only after a time-out takes place in the upper level software. By determining that the packet was lost at the driver level, a transmit command can be reissued to the AT/LANTIC Controller once the overflow routine is completed (as in step 11). Also, it is possible for the AT/LANTIC Controller to defer indefinitely, when it is stopped on a busy network. Step 5 also alleviates this problem. Step 5 is essential and should not be omitted from the overflow routine, in order for the AT/LANTIC Controller to operate correctly.

6. Place the AT/LANTIC Controller in either mode 1 or mode 2 loopback. This can be accomplished by setting bits D2 and D1, of the Transmit Configuration Register, to 0,1 or 1,0 respectively.

7. Issue the START command to the AT/LANTIC Controller. This can be accomplished by Writing 22H to the Command Register. This is necessary to activate the AT/LANTIC Controller's Remote DMA channel.

8. Remove one or more packets from the receive buffer ring.

9. Reset the overwrite warning (OVW, overflow) bit in the Interrupt Status Register.

10. Take the AT/LANTIC Controller out of loopback. This is done by Writing the Transmit Configuration Register with the value it contains during normal operation. (Bits D2 and D1 should both be programmed to 0.)

11. If the "Resend" variable is set to a 1, reset the "Resend" variable and reissue the transmit command. This is done by writing a value of 26H to the Command Register. If the "Resend" variable is 0, nothing needs to be done.

**Note 1:** If Remote DMA is not being used, the AT/LANTIC Controller does not need to be started before packets can be removed from the receive buffer ring. Hence, step 8 could be done before step 7, eliminating or reducing the time spent polling in step 5.

**Note 2:** When the AT/LANTIC Controller is in STOP mode, the Missed Packet Tally counter is disabled.

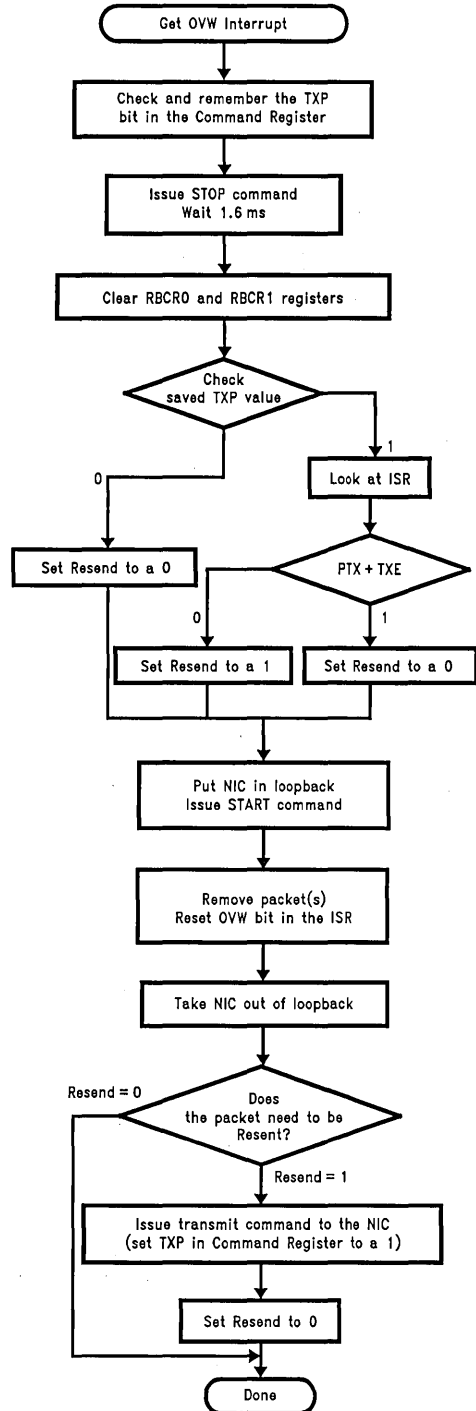


FIGURE 32. Overflow Routine

TL/F/11498-29

## 6.0 Operation of AT/LANTIC Controller (Continued)

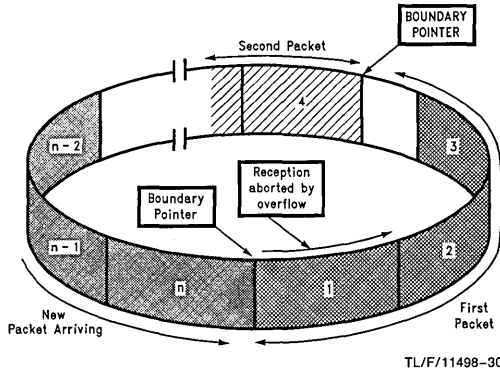


FIGURE 33. Received Packet Aborted if it Hits Boundary

### Enabling the AT/LANTIC Controller on an Active Network

After the AT/LANTIC Controller has been initialized the procedure for disabling and then re-enabling the AT/LANTIC Controller on the network is similar to handling Receive Buffer Ring overflow as described previously.

1. Program Command Register for page 0 (Command Register = 21H)
2. Initialize Data Configuration Register (DCR)
3. Clear Remote Byte Count Registers (RBCR0, RBCR1) if using Remote DMA
4. Initialize Receive Configuration Register (RCR)
5. Place the AT/LANTIC Controller in LOOPBACK mode 1 or 2 (Transmit Configuration Register = 02H or 04H)
6. Initialize Receive Buffer Ring: Boundary Pointer (BNDRY), Page Start (PSTART), and Page Stop (PSTOP)
7. Clear Interrupt Status Register (ISR) by writing 0FFH to it
8. Initialize Interrupt Mask Register (IMR)
9. Program Command Register for page 1 (Command Register = 61H)
  - i) Initialize Physical Address Registers (PAR0-PAR5)
  - ii) Initialize Multicast Address Registers (MAR0-MAR7)
  - iii) Initialize CURRENT pointer
10. Put AT/LANTIC Controller in START mode (Command Register = 22H). The local receive DMA is still not active since the AT/LANTIC Controller is in LOOPBACK.
11. Initialize the Transmit Configuration for the intended value. The AT/LANTIC Controller is now ready for transmission and reception.

### End of Packet Operations

At the end of the packet the AT/LANTIC Controller determines whether the received packet is to be accepted or rejected. It either branches to a routine to store the Buffer Header or to another routine that recovers the buffers used to store the packet.

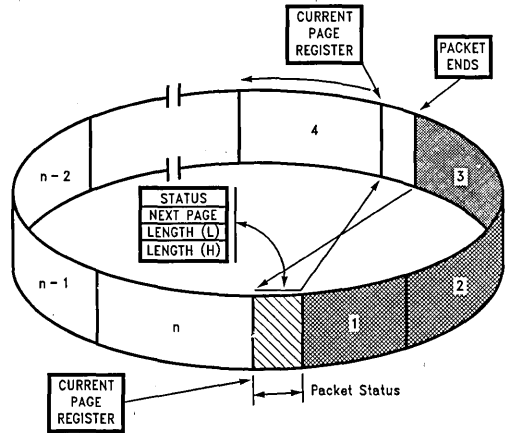


FIGURE 34. Termination of Received Packet—Packet Accepted

### Successful Reception

If the packet is successfully received, the DMA is restored to the first buffer used to store the packet (pointed to by the Current Page Register). The DMA then stores the Receive Status, a Pointer to where the next packet will be stored (Buffer 4) and the number of received bytes. Note that the remaining bytes in the last buffer are discarded and reception of the next packet begins on the next empty 256-byte buffer boundary. The Current Page Register is then initialized to the next available buffer in the Buffer Ring. (The location of the next buffer had been previously calculated and temporarily stored in an internal scratchpad register.)

### Buffer Recovery for Rejected Packets

If the packet is a runt packet or contains CRC or Frame Alignment errors, it is rejected. The buffer management logic resets the DMA back to the first buffer page used to store the packet (pointed to by CURR), recovering all buffers that had been used to store the rejected packet. This operation will not be performed if the AT/LANTIC Controller is programmed to accept either runt packets or packets with CRC or Frame Alignment errors. The received CRC is always stored in buffer memory after the last byte of received data for the packet.

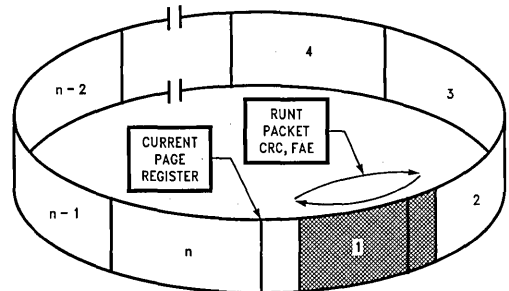


FIGURE 35. Termination of Receive Packet—Packet Reject

## 6.0 Operation of AT/LANTIC Controller (Continued)

### Error Recovery

If the packet is rejected as shown, the DMA is restored by the AT/LANTIC Controller by reprogramming the DMA starting address pointed to by the Current Page Register.

### Storage Format for Received Packets

The following diagrams describe the format for how received packets are placed into memory by the local DMA channel. These modes are selected in the Data Configuration Register.

AD15	AD8	AD7	AD0
Next Packet Pointer		Receive Status	
Receive Byte Count 1		Receive Byte Count 0	
Byte 2		Byte 1	

BOS = 0, WTS = 1 in Data Configuration Register. This format is used with Series 32xxx, or 808xx processors.

AD15	AD8	AD7	AD0
Next Packet Pointer		Receive Status	
Receive Byte Count 0		Receive Byte Count 1	
Byte 1		Byte 2	

BOS = 1, WTS = 1 in Data Configuration Register. This format is used with 680x0 type processors. (Note: The Receiver Count ordering remains the same for BOS = 0 or 1.)

Receive Status
Next Packet Pointer
Receive Byte Count 0
Receive Byte Count 1
Byte 0
Byte 1

BOS = 0, WTS = 0 in Data Configuration Register. This format is used with general 8-bit processors.

### 6.4 PACKET TRANSMISSION

The Local DMA is also used during transmission of a packet. Three registers control the DMA transfer during transmission, a Transmit Page Start Address Register (TPSR) and the Transmit Byte Count Registers (TBCR0,1). When the AT/LANTIC Controller receives a command to transmit the packet pointed to by these registers, buffer memory data will be moved into the FIFO as required during transmission. The AT/LANTIC Controller will generate and append the preamble, synch and CRC fields.

#### General Transmit Packet Format

Transmit	Destination Address	6 Bytes
Byte	Source Address	6 Bytes
Count	Type/Length	2 Bytes
TBCR0, 1	Data	≥ 46 Bytes
	Pad (If data < 46 Bytes)	

### Transmit Packet Assembly

The AT/LANTIC Controller requires a contiguous assembled packet with the format shown. The transmit byte count includes the Destination Address, Source Address, Length Field and Data. It does not include preamble and CRC.

When transmitting data smaller than 46 bytes, the packet must be padded to a minimum size of 64 bytes. The programmer is responsible for adding and stripping pad bytes. The packets are placed in the buffer RAM by the system. In I/O Mode the system programs the NIC Core's Remote DMA to mode the data from the data port to the RAM handshaking with system transfers loading the I/O data port. In Shared Memory Mode the packets are written directly to the RAM by system using standard memory transfer instructions (MOV).

For I/O mode the data transfer must be 16 bits (1 word) when in 16-bit mode, and 8 bits when the AT/LANTIC Controller is set in 8-bit mode. The data width is selected by setting the WTS bit in the Data Configuration Register and setting the DWID pin for the proper mode.

In Shared Memory mode data transfer can be accomplished by using either 8- or 16-bit data transfer instructions, because this mode responds to 8/16-bit data signalling on the ISA bus. In this mode Shared Memory Control Register 2-bit 6 sets the bus interface data width, and the NIC Core's data width is set by the WTS bit in the Data Configuration Register.

### Transmission

Prior to transmission, the TPSR (Transmit Page Start Register) and TBCR0, TBCR1 (Transmit Byte Count Registers) must be initialized. To initiate transmission of the packet the TXP bit in the Command Register is set. The Transmit Status Register (TSR) is cleared and the AT/LANTIC Controller begins to prefetch transmit data from memory (unless the AT/LANTIC Controller is currently receiving). If the inter-frame gap has timed out the AT/LANTIC Controller will begin transmission.

### Conditions Required to Begin Transmission

In order to transmit a packet, the following three conditions must be met:

1. The Interframe Gap Timer has timed out the first 6.4 μs of the Interframe Gap
2. At least one byte has entered the FIFO. (This indicates that the burst transfer has been started)
3. If a collision had been detected then before transmission the packet time must have timed out.

In typical systems the AT/LANTIC Controller prefetches the first burst of bytes before the 6.4 μs timer expires. The time during which AT/LANTIC Controller transmits preamble can also be used to load the FIFO.

Note: If carrier sense is asserted before a byte has been loaded into the FIFO, the AT/LANTIC Controller will become a receiver.

### Collision Recovery

During transmission, the Buffer Management logic monitors the transmit circuitry to determine if a collision has occurred. If a collision is detected, the Buffer Management logic will reset the FIFO and restore the Transmit DMA pointers for retransmission of the packet. The COL bit will be set in the TSR and the NCR (Number of Collisions Register) will be incremented. If 15 retransmissions each result in a collision the transmission will be aborted and the ABT bit in the TSR will be set.

Note: NCR reads as zeroes if excessive collisions are encountered.

## 6.0 Operation of AT/LANTIC Controller (Continued)

### Transmit Packet Assembly Format

The following diagrams describe the format for how packets must be assembled prior to transmission for different byte ordering schemes. The various formats are selected in the Data Configuration Register.

D15	D8	D7	D0
Destination Address 1		Destination Address 0	
Destination Address 3		Destination Address 2	
Destination Address 5		Destination Address 4	
Source Address 1		Source Address 0	
Source Address 3		Source Address 2	
Source Address 5		Source Address 4	
Type/Length 1		Type Length 0	
Data 1		Data 0	

BOS = 1, WTS = 1 in Data Configuration Register.

This format is used with Series 32xxx, or 808xx processors.

D15	D8	D7	D0
Destination Address 0		Destination Address 1	
Destination Address 2		Destination Address 3	
Destination Address 4		Destination Address 5	
Source Address 0		Source Address 1	
Source Address 2		Source Address 3	
Source Address 4		Source Address 5	
Type/Length 0		Type Length 1	
Data 0		Data 1	

BOS = 1, WTS = 1 in Data Configuration Register.

This format is used with 680x0 type processors.

D7	D0
Destination Address 0	
Destination Address 1	
Destination Address 2	
Destination Address 3	
Destination Address 4	
Destination Address 5	
Source Address 0	
Source Address 1	
Source Address 2	
Source Address 3	
Source Address 4	
Source Address 5	

BOS = 0, WTS = 0 in a Data Configuration Register.

This format is used with 8-bit processors.

**Note:** All examples above will result in a transmission of a packet in order of DA0, DA1, DA3 . . . bits within each byte will be transmitted least significant bit first.

DA = Destination Address

### 6.5 LOOPBACK DIAGNOSTICS

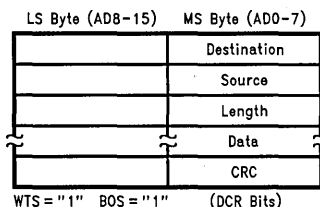
Three forms of local loopback are provided on the AT/LANTIC Controller. The user has the ability to loopback through the deserializer on the controller, through the ENDEC module or transceiver. **Because of the half duplex architecture of the AT/LANTIC Controller, loopback testing is a special mode of operation with the following restrictions:**

#### Restrictions during Loopback

The FIFO is split into two halves, one half is used for transmission the other for reception. Only 8-bit fields can be fetched from memory so two tests are required for 16-bit systems to verify integrity of the entire data path. During loopback the maximum latency to obtain access to the buffer memory is 2.0  $\mu$ s. Systems that wish to use the loopback test yet do not meet this latency can limit the loopback packet to 7 bytes without experiencing underflow. Only the last 8 bytes of the loopback packet are retained in the FIFO. The last 8 bytes can be read through the FIFO register which will advance through the FIFO to allow reading the receive packet sequentially.

Destination Address	= 6 bytes Station Physical Address
Source Address	= 6 bytes Station Physical Address
Length	2 bytes
Data	= 46 to 1500 bytes
CRC	Appended by AT/LANTIC Controller if CRC = 0 in TCR

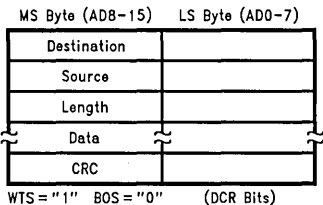
When in word-wide mode with Byte Order Select set, the loopback packet must be assembled in the even byte locations as shown below. (The loopback only operated with byte wide transfers.)



TL/F/11498-64

## 6.0 Operation of AT/LANTIC Controller (Continued)

When in word-wide mode with Byte Order Select low, the following format must be used for the loopback packet.



TL/F/11498-65

**Note:** When using loopback in word mode 2n bytes must be programmed in the TBCRO, 1. When n = actual number of bytes assembled in even or odd location.

To initiate a loopback the user first assembles the loopback packet then selects the type of loopback using the Transmit Configuration register bits LB0, LB1. The transmit configuration register must also be set to enable or disable CRC generation during transmission. The user then issues a normal transmit command to send the packet. During loopback the receiver checks for an address match and if CRC bit in the TCR is set, the receiver will also check the CRC. The last 8 bytes of the loopback packet are buffered and can read out of the FIFO using FIFO read port.

### Loopback Modes

**MODE1:** Loopback through the AT/LANTIC Controller Module (LB1 = 0, LB0 = 1): If this loopback is used, the AT/LANTIC Controller Module's serializer is connected to the deserializer.

**MODE 2:** Loopback through the ENDEC Module (LB1 = 1, LB0 = 0): If the loopback is to be performed through the SNI, the AT/LANTIC Controller provides a control (LPBK) that forces the ENDEC module to loopback all signals.

**MODE 3:** Loopback to the external coax interface or twisted pair interface module (LB1 = 1, LB0 = 1). Packets can be transmitted to the cable in loopback mode to check all of the transmit and receive paths and the cable itself. If, in twisted pair mode, there is a link fail the transmitter will be disabled which could give misleading results in Mode 3. The link integrity should be checked, by reading Configuration Register B, before this test.

**Note:** Collision and Carrier Sense can be generated by the ENDEC module and are masked by the NIC module. It is not possible to go directly between the loopback modes, it is necessary to return to normal operation (OOH) when changing modes.

### Reading the Loopback Packet

The last eight bytes of a received packet can be examined by 8 consecutive reads of the FIFO register. The FIFO pointer is increment after the rising edge of the CPU's read strobe by internally synchronizing and advancing the pointer. This may take up to four bus clock cycles, if the pointer has not been incremented by the time the CPU reads the FIFO register again, the AT/LANTIC Controller will insert wait states.

**Note:** The FIFO may only be read during Loopback. Reading the FIFO at any other time will cause the AT/LANTIC Controller to malfunction.

### Alignment of the Received Packet in the FIFO

Reception of the packet in the FIFO begins at location zero, after the FIFO pointer reaches the last location in the FIFO, the pointer wraps to the top of the FIFO overwriting the previously received data.

This process is continued until the last byte is received. The AT/LANTIC Controller then appends the received byte count in the next two locations of the FIFO. The contents of the Upper Byte Count are also copied to the next FIFO location. The number of bytes used in the loopback packet determined the alignment of the packet in the FIFO. The alignment for a 64-byte packet is shown below.

FIFO Location	FIFO Contents	
0	Lower Byte Count	→ First Byte Read
1	Upper Byte count	→ Second Byte Read
2	Upper Byte Count	•
3	Last Byte	•
4	CRC1	•
5	CRC2	•
6	CRC3	•
7	CRC4	→ Last Byte Read

For the following alignment in the FIFO the packet length should be  $(N \times 8) + 5$  Bytes. Note that if the CRC bit in the TCR is set, CRC will not be appended by the transmitter. If the CRC is appended by the transmitter, the 1st four bytes, bytes N-3 to N, correspond to the CRC.

FIFO Location	FIFO Contents	
0	Byte N-4	→ First Byte Read
1	Byte N-3 (CRC1)	→ Second Byte Read
2	Byte N-2 (CRC2)	•
3	Byte N-1 (CRC3)	•
4	Byte N (CRC4)	•
5	Lower Byte Count	•
6	Upper Byte Count	•
7	Upper Byte Count	→ Last Byte Read

### Loopback Tests

Loopback capabilities are provided to allow certain tests to be performed to validate operation of the AT/LANTIC Controller prior to transmitting and receiving packets on a live network. Typically these tests may be performed during power up of a node. The diagnostic provides support to verify the following:

1. Verify integrity of data path. Received data is checked against transmitted data.
2. Verify CRC logic's capability to generate good CRC on transmit, verify CRC on receive (good or bad CRC).
3. Verify that the Address Recognition Logic can
  - a. Recognize address match packets
  - b. Reject packets that fail to match an address

### Loopback Operation in the AT/LANTIC Controller

Loopback is a modified form of transmission using only half of the FIFO. This places certain restrictions on the use of

## 6.0 Operation of AT/LANTIC Controller (Continued)

loopback testing. When loopback mode is selected in the TCR, the FIFO is split. A packet should be assembled in memory with programming of TPSR and TBCR0, TBCR1 registers. When the transmit command is issued the following operations occur:

### TRANSMITTER ACTIONS

1. Data is transferred from memory by the DMA until the FIFO is filled. For each transfer TBCR0 and TBCR1 are decremented. (Subsequent burst transfers are initiated when the number of bytes in the FIFO drops below the programmed threshold.)
2. The AT/LANTIC Controller generates 56 bits of preamble followed by an 8-bit synch pattern.
3. Data transferred from FIFO to serializer.
4. If CRC = 1 in TCR, no CRC calculated by AT/LANTIC Controller, the last byte transmitted is the last byte from the FIFO (allows software CRC to be appended). If CRC = 0, AT/LANTIC Controller calculates and appends four bytes of CRC.
5. At end of Transmission PTX bit set in ISR.

### RECEIVER ACTIONS

1. Wait for synch, all preamble stripped.
2. Store packet in FIFO, increment receive byte count for each incoming byte.
3. If CRC = 0 in TRC, receiver checks incoming packet for CRC errors. If CRC = 1 in TCR, receiver does not check CRC errors, CRC error bit always set in RSR (for address matching packets).
4. At end of receive, receive byte count written into FIFO, receive status register is updated. The PRX bit is typically set in the RSR even if the address does not match. If CRC errors are forced, the packet must match the address filters in order for the CRC error bit in the RS to be set.

### EXAMPLES

The following examples show what results can be expected from a properly operating AT/LANTIC Controller during loopback. The restrictions and results of each type of loopback are listed for reference. The loopback tests are divided into two sets of tests. One to verify the data path, CRC generation and byte count through all three paths. The second set of tests uses internal loopback to verify the receiver's CRC checking and address recognition. For all of the tests the DCR was programmed to 40H.

Path	TCR	RCR	TSR	RSR	ISR
AT/LANTIC Controller Internal	02	1F	53 (Note 1)	02 (Note 2)	02 (Note 3)

**Note 1:** Since carrier sense and collision detect are generated in the ENDEC module. They are blocked during internal loopback, carrier and CD heartbeat are not seen and the CRS and CDH bits are set.

**Note 2:** CRC errors are always indicated by receiver if CRC is appended by the transmitter.

**Note 3:** Only the PTX bit in the ISR is set, the PRX bit is only set if status is written to memory. In loopback this action does not occur and the PRX bit remains 0 for all loopback modes.

**Note 4:** All values are hex.

Path	TCR	RCR	TSR	RSR	ISR
AT/LANTIC Controller Internal	04	1F	43 (Note 1)	02	02

**Note 1:** CDH is set, CRS is not set since it is generated by the external encoder/decoder.

Path	TCR	RCR	TSR	RSR	ISR
AT/LANTIC Controller External	06	1F	03 (Note 1)	02	02 (Note 2)

**Note 1:** CDH and CRS should not be set. The TSR however, could also contain 01H,03H,07H and a variety of other values depending on whether collisions were encountered or the packet was deferred.

**Note 2:** Will contain 08H if packet is not transmittable.

**Note 3:** During external loopback the AT/LANTIC Controller is now exposed to network traffic, it is therefore possible for the contents of both the Receive portion of the FIFO and the RSR to be corrupted by any other packet on the network. Thus in a live network the contents of the FIFO and RSR should not be depended on. The AT/LANTIC Controller will still abide by the standard CSMA/CD protocol in external loopback mode (i.e. the network will not be disturbed by the loopback packet).

**Note 4:** All values are hex.

### CRC and Address Recognition

The next three tests exercise the address recognition logic and CRC. These tests should be performed using internal loopback only so that the AT/LANTIC Controller is isolated from interference from the network. These tests also require the capability to generate CRC in software.

The address recognition logic cannot be directly tested. The CRC and FAE bits in the RSR are only set if the address in the packet matches the address filters. If errors are expected to be set and they are not set, the packet has been rejected on the basis of an address mismatch. The following sequence of packets will test the address recognition logic. The DCR should be set to 40H, the TCR should be set to 03H with a software generated CRC.

Packet Contents		Results	
Test	Address	CRC	RSR
Test A	Matching	Good	01 (Note 1)
Test B	Matching	Bad	02 (Note 2)
Test C	Non-Matching	Bad	01

**Note 1:** Status will read 21H if multicast address used.

**Note 2:** Status will read 22H if multicast address used.

**Note 3:** In test A, the RSR is set up. In test B the address is found to match since the CRC is flagged as bad. Test C proves that the address recognition logic can distinguish a bad address and does not notify the RSR of the bad CRC. The receiving CRC is proven to work in test A and test B.

**Note 4:** All values are hex.



## 6.0 Operation of AT/LANTIC Controller (Continued)

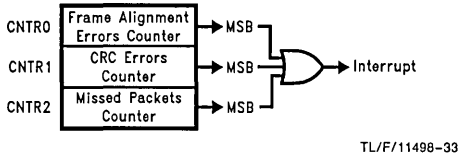


FIGURE 36. Tally Counters

### Network Management Functions

Network management capabilities are required for maintenance and planning of a local area network. The AT/LANTIC Controller supports the minimum requirement for network management in hardware, the remaining requirements can be met with software. Software alone can not track during reception of packets: CRC errors, Frame Alignment errors, and missed packets, *Figure 36*.

Since errored packets can be rejected, the status associated with these packets is lost unless the CPU can access the Receive Status Register before the next packer arrives. In situations where another packet arrives very quickly, the CPU may have no opportunity to do this. The AT/LANTIC Controller counts the number of packets with CRC errors and Frame Alignment errors. 8-bit counters have been selected to reduce overhead. The counters will generate interrupts whenever their MSBs are set so that a software routine can accumulate the network statistics and reset the counter before overflow occurs. The counters are sticky so that when they reach a count of 192 (COH) counting is halted. An additional counter is provided to count the number of packets the AT/LANTIC Controller misses due to buffer overflow or being offline.

The structure of the counters is shown in *Figure 36*.

Additional information required for network management is available in the Receive and Transmit Status Registers. Transmit status is available after each transmission for information regarding events during transmission.

Typically, the following statistics might be gathered in software:

- Traffic:
  - Frames Sent OK
  - Frames Received OK
  - Multicast Frames Received
  - Packets Lost Due to Lack of Resources
  - Retries/Packet
- Errors:
  - CRC Errors
  - Alignment Errors
  - Excessive Collisions
  - Packet with Length Errors
  - Heartbeat Failure

### 6.6 MEMORY ARBITRATION AND BUS OPERATION

The AT/LANTIC Controller will always operate as a slave device on its peripheral interface to the ISA bus. However on the memory bus, the AT/LANTIC Controller operates in three possible modes:

1. Bus Master of Local Packet Buffer RAM
2. Bus Slave when accessed by the CPU via the Bus Interface
3. Idle, when no activity is occurring.

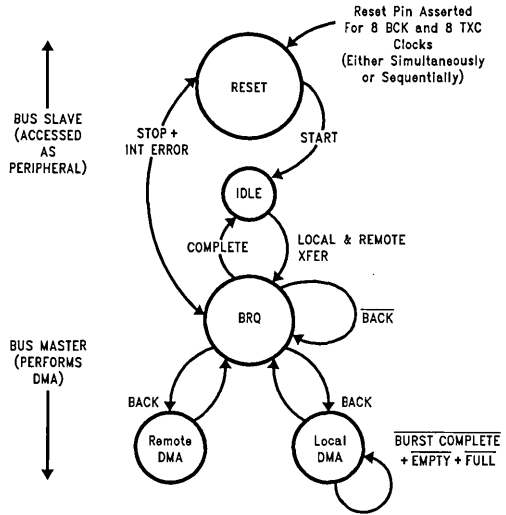


FIGURE 37. DP8390 Core Bus States

Upon power-up the AT/LANTIC Controller is in an indeterminate state. After receiving a hardware reset the AT/LANTIC Controller is a bus slave in the Reset State, the receiver and transmitter are both disabled in this state. The reset state can be re-entered under four conditions, soft reset (Stop Command), register reset (reset port in I/O mode, bit in Control Register 1 in shared memory mode), hard reset (RESET input) or an error that shuts down the receiver or transmitter (FIFO underflow or overflow, receive buffer ring overflow).

After initialization of registers, the AT/LANTIC Controller is issued a Start command and the AT/LANTIC Controller enters Idle state. Until the DMA is required the AT/LANTIC Controller remains in idle state.

The idle state is exited and the AT/LANTIC Controller will drive the local memory bus when a request from the FIFO in the DP8390 (NIC) core causes the memory bus interface logic to issue a read or write operation, such as when the AT/LANTIC Controller is transmitting or receiving data.

In I/O mode the NIC Core's Remote DMA also requests access from the memory bus. When software programs an I/O mode data transfer between the CPU and the buffer RAM, the Remote DMA controls this request.

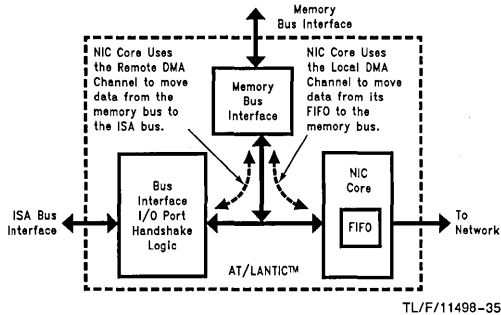
In Shared Memory Mode, the memory bus is accessed via the CPU interface directly.

All Local DMA transfers are burst transfers, the DMA will transfer an exact burst of bytes programmed in the Data Configuration Register (DCR) then relinquish the memory bus. If there are remaining bytes in the FIFO the next burst will not be initiated until the FIFO threshold is exceeded.

#### I/O Mode Operation

In I/O mode the AT/LANTIC Controller transfers data to and from the packet buffer RAM by utilizing the Remote DMA logic which is programmed by the main system CPU to transfer data through the AT/LANTIC Controller's internal data port register.

## 6.0 Operation of AT/LANTIC Controller (Continued)



**FIGURE 38. I/O Operation: All Data Transfers and Arbitration is Controlled by the NIC Core**

### INTERLEAVED LOCAL/REMOTE OPERATION

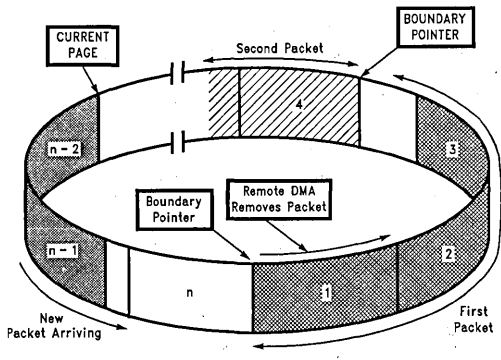
When in I/O mode the remote DMA is used to transfer data to/from the main system. If a remote DMA transfer is initiated or in progress when a packet is being received or transmitted, the Remote DMA transfers will be interrupted for higher priority Local DMA transfers. When the Local DMA transfer is completed the Remote DMA will re-arbitrate for the bus and continue its transfers.

If the FIFO requires service while a remote DMA is in progress the Local DMA burst is appended to the Remote Transfer. When switching from a local transfer to a remote transfer there is a break to allow the CPU to fairly contend for the bus.

### REMOTE DMA BI-DIRECTIONAL PORT

The Remote DMA transfers data between the local buffer memory and the internal bidirectional port (memory to I/O transfer).

This transfer is arbitrated on a transfer by transfer basis versus the burst transfer mode used for Local DMA transfers. This bidirectional port is integrated onto the AT/LANTIC Controller, and is read/written by the host. All transfers through this port are asynchronous. At any one time transfers are limited to one direction, either from the port to local buffer memory (Remote Write) or from local buffer memory to the port (Remote Read).



**FIGURE 39. 1st Received Packet Removed by Remote DMA**

### I/O MODE REMOVING PACKETS FROM RING

Network activity is isolated on a local bus, where the AT/LANTIC Controller's local DMA channel performs burst transfers between the buffer memory and the AT/LANTIC Controller's FIFO. The Remote DMA transfers data between the buffer memory and the host memory via the internal bidirectional I/O port. The Remote DMA provides local addressing capability and is used as a slave DMA by the host. The host system reads the I/O port to transfer data between the system and I/O port. The AT/LANTIC Controller allows Local and Remote DMA operations to be interleaved.

Packets are removed from the ring using the Remote DMA. When using the Remote DMA the Send Packet command can be used. This programs the Remote DMA to automatically remove the received packet pointed to by the Boundary Pointer. At the end of the transfer, the AT/LANTIC Controller moves the Boundary Pointer, freeing additional buffers for reception. The Boundary Pointer can also be moved manually by programming the Boundary Register. Care should be taken to keep the Boundary Pointer at least one buffer behind the Current Page Pointer.

Data transfer by the Remote DMA to the integrated I/O data port is dependent on whether the AT/LANTIC Controller is set into 8-bit mode or 16-bit mode. In 8-bit mode all transfers are 8 bits (1 byte) wide. When in 16-bit mode all transfers are 16 bits (1 word) wide. The data width is selected by setting the WTS bit in the Data Configuration Register and setting the DWID pin for the proper mode.

The following is a suggested method for maintaining the Receive Buffer Ring pointers if in shared memory mode or if remote read is used in I/O mode.

1. At initialization, set up a software variable (`next_pkt`) to indicate where the next packet will be read. At the beginning of each Remote Read DMA operation, the value of `next_pkt` will be loaded into RSAR0 and RSAR1.
2. When initializing the AT/LANTIC Controller set:
 

```
BNDRY = PSTART
CURR = PSTART + 1
next_pkt = PSTART + 1
```
3. After a packet is DMAed from the Receive Buffer Ring, the Next Page Pointer (second byte in the AT/LANTIC Controller buffer header is used to update BNDRY and `next_pkt`.
 

```
next_pkt = Next Page Pointer
BNDRY = Next Page Pointer - 1
```

If  $BNDRY < PSTART$  then  $BNDRY = PSTOP - 1$

Note the size of the Receive Buffer Ring is reduced by one 256 byte buffer, this will not, however, impede the operation of the AT/LANTIC Controller. The advantage of this scheme is that it easily differentiates between buffer full and buffer empty; it is full if  $BNDRY = CURR$ ; empty when  $BNDRY = CURR - 1$ . If, in I/O mode, send packet is used to empty the buffer ring this scheme cannot be used. BNDRY must be initialized equal to CURR, or the first executed send packet will not return data from the received packet, which will be written at CURR. The Overwrite Warning bit of the Interrupt Status Register must be used in this mode to differentiate between buffer full and buffer empty.

## 6.0 Operation of AT/LANTIC Controller (Continued)

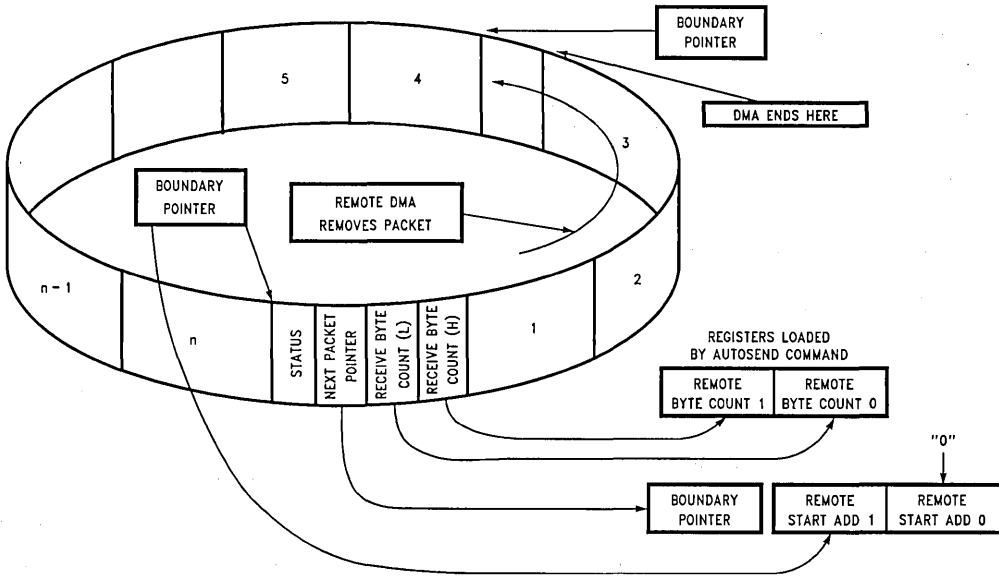


FIGURE 40. Remote DMA Autoinitialization from Buffer Ring

TL/F/11498-37

### I/O MODE REMOTE DMA COMMANDS

The Remote DMA channel is used in the I/O Mode to both assemble packets for transmission, and to remove received packets from the Receive Buffer Ring. It may also be used for moving blocks of data or commands between host memory and local buffer memory. (In Shared Memory Mode, the Remote DMA should be disabled, and not used. Packet transfer to/from the system is accomplished by normal CPU read/write operations.)

There are three modes of Remote DMA operation: Remote Write, Remote Read, or Send Packet.

Two register pairs are used to control the Remote DMA, a Remote Start Address (RSAR0, RSAR1) and a Remote Byte Count (RBCR0, RBCR1) register pair. The Start Address Register pair points to the beginning of the block to be moved while the Byte Count Register pair is used to indicate the number of bytes to be transferred. Full handshake logic is provided to move data between local buffer memory and a bidirectional I/O port.

**Remote Write:** A Remote Write transfer is used to move a block of data from the host into local buffer memory. The Remote DMA will read data from the I/O port and sequentially write it to local buffer memory beginning at the Remote Start Address. The DMA Address will be incremented and the Byte Counter will be decremented after each transfer. The DMA is terminated when the Remote Byte Count Register reaches a count of zero.

**Remote Read:** A Remote Read transfer is used to move a block of data from local buffer memory to the host. The Remote DMA will sequentially read data from the local buffer memory, beginning at the Remote Start Address, and write data to the I/O port. The DMA Address will be incremented and the Byte Counter will be decremented after each transfer. The DMA is terminated when the Remote Byte Count Register reaches zero.

**Send Packet Command:** The Remote DMA channel can be automatically initialized to transfer a single packet from the Receive Buffer Ring. The CPU begins this transfer by issuing a "Send Packet" Command. The DMA will be initialized to the value of the Boundary Pointer Register and the Remote Byte Count Register pair (RBCR0, RBCR1) will be initialized to the value of the Receive Byte Count fields found in the Buffer Header of each packet. After the data is transferred, the Boundary Pointer is advanced to allow the buffers to be used for new receive packets. The Remote Read will terminate when the Byte Count equals zero. The Remote DMA is then prepared to read the next packet from the Receive Buffer Ring. If the DMA pointer crosses the Page Stop Register, it is reset to the Page Start Address. This allows the Remote DMA to remove packets that have wrapped around to the top of the Receive Buffer Ring.

**Note 1:** In order for the AT/LANTIC Controller to correctly execute the Send Packet command, the upper Remote Byte Count Register (RBCR1) must first be loaded with 0FH.

**Note 2:** The Send Packet command cannot be used with 680x0 type processors.

### I/O MODE READ TIMING

1. The DMA reads a word from local buffer memory and writes the word into the internal latch, increments the DMA address and decrements the byte count (RBCR0,1).
2. Internally a request line is asserted to enable the system to read the port. If the system reads this port before the data has been written, then the system is sent a wait signal to wait until the data has been written to the port. Once written the system's read is allowed to complete.
3. The system reads the port, the read strobe for the port is used as an acknowledgement to the Remote DMA and it goes back to step 1.

## 6.0 Operation of AT/LANTIC Controller (Continued)

Steps 1–3 are repeated until the remote DMA is complete (i.e. the byte count has gone to zero).

Note that in order for the Remote DMA to transfer a word from memory to the latch, it must arbitrate access to the local buffer RAM. After each word is transferred to the internal latch, access to the RAM is relinquished. If a Local DMA is in progress, the Remote DMA is held off until the local DMA is complete.

### I/O MODE WRITE TIMING

A Remote Write operation transfers data from the I/O port to the local buffer RAM. The system transfers a byte-word to the latch via  $\overline{IOWR}$ . This write strobe is detected by the AT/LANTIC Controller and the byte/word is transferred to local buffer memory. The Remote DMA holds off further transfers into the latch until the current byte/word has been transferred from the latch.

1. AT/LANTIC Controller awaits data to be written by the system. System writes byte/word into latch.
2. Remote DMA reads contents of port and writes byte/word to local buffer memory, increments address and decrements byte count (RBCRO,1).
3. Go back to step 1.

Steps 1–3 are repeated until the remote DMA is complete.

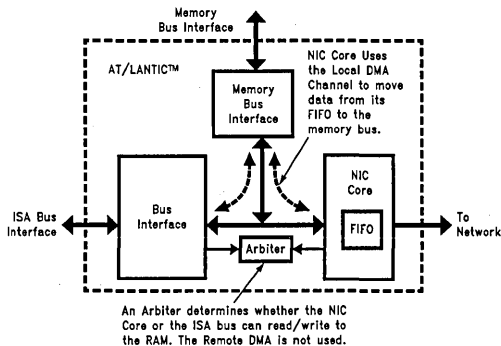


FIGURE 41. Shared Memory Mode the ISA Bus Directly Access the RAM

### Shared Memory Mode Operation

In shared memory mode the AT/LANTIC Controller transfers data to or from the packet buffer RAM directly from or to the ISA bus. The buffer RAM is mapped into system memory with the AT/LANTIC Controller doing all address decoding, synchronization and handshaking.

### INTERLEAVED SHARED/LOCAL OPERATION

When in shared memory mode the local DMA is used to transfer data to or from the FIFO in the NIC core and ultimately the network. If a local DMA transfer is in progress when a shared memory access occurs the system is sent a wait state signal until the local DMA has been completed. If the shared memory access begins first then it will be completed before any local DMA is allowed.

### SHARED MEMORY HOST DATA TRANSFER

In Shared Memory Mode the system reads data from the RAM directly, usually using memory string move instructions. The memory is enabled by setting D6 of Shared Memory Control Register 1. The base address of the memory is programmed by writing to the Control Registers.

If DWID is low only Control Register 1 is used to program base address, so the memory must exist in the lower 1 Mbytes of system memory. A19 is always compared to a 1 when DWID is low. The A13–18 bits are compared to the address lines, if there is 8k of memory. A13 is not compared in 16k mode, A13–14 are not compared in 32k mode (8-bit non-compatible) and A13–15 are not compared in 64k mode (16-bit non-compatible).

If DWID is high both Control Registers must be programmed to set the base address, so the memory can exist anywhere in up to 16 Gbytes of system memory. LA19 can be either 1 or 0. The same limited decode, as detailed above, also occurs depending on the memory size.

### SHARED MEMORY READ TIMING

The system executes a normal memory read cycle which the AT/LANTIC Controller will complete immediately, if idle, or insert wait states into if local DMA is current. The byte or word of data is fetched from the buffer RAM via the memory support bus.

### SHARED MEMORY WRITE TIMING

The system executes a normal memory write cycle which the AT/LANTIC Controller will complete immediately, if idle, or insert wait states into if local DMA is current. The byte or word of data is written to the buffer RAM via the memory support bus.

## 6.7 FUNCTIONAL BUS TIMING

This section describes the bus cycles that the AT/LANTIC Controller performs. These timings can be subdivided into 3 basic categories:

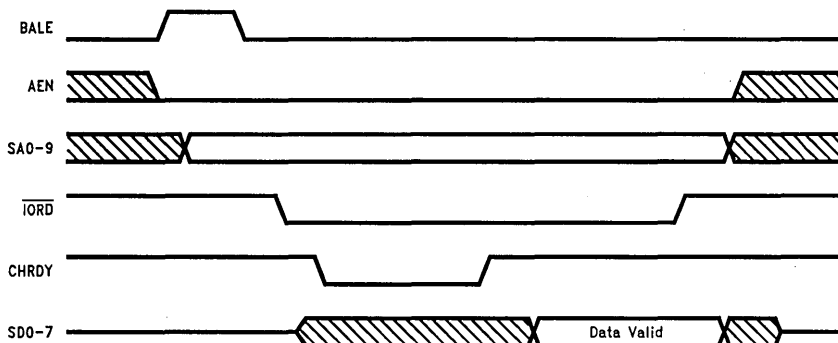
1. ISA I/O Access: There are register accesses in both modes, and I/O data accesses in I/O mode.
2. Shared RAM ISA Accesses: These are the timing for the ISA bus accesses through the AT/LANTIC Controller to the memory bus and buffer RAM.
3. Boot PROM ISA Accesses: These are the timing for the ISA bus accesses through the AT/LANTIC Controller to the memory bus and boot PROM.
4. Local and I/O RAM Accesses: This is the timing of the local DMA, accesses from the NIC Core FIFO to the RAM, and the Remote DMA accesses to the RAM over the memory bus.

### ISA Bus I/O Accesses

The AT/LANTIC Controller is designed to directly interface to the ISA bus (PC-AT backplane bus). The CPU can read or write any internal registers. All register accesses are byte wide. The functional timing for AT/LANTIC Controller accesses are shown in the following pages.

## 6.0 Operation of AT/LANTIC Controller (Continued)

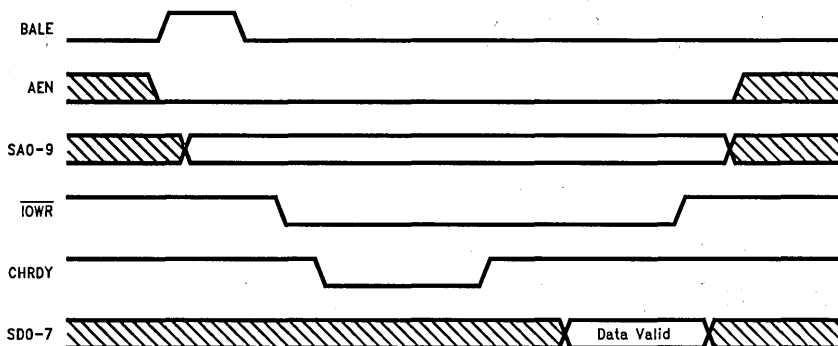
8-Bit I/O Mode Slave Read



TL/F/11498-39

This is the type of cycle used to read from a register or, in 8-bit I/O mode, from a data transfer port. These accesses are entirely asynchronous, with the AT/LANTIC Controller responding when it decodes the correct address on SA0-9 and an  $\overline{\text{IORD}}$ . If AEN is high the cycle will be ignored. CHRDY is deasserted if the AT/LANTIC Controller is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted. The data will always appear on SD0-7.

8-Bit I/O Mode Slave Write

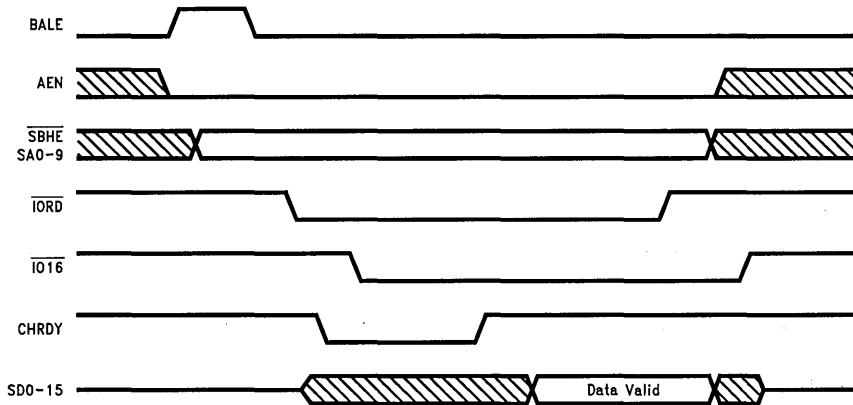


TL/F/11498-40

This is the type of cycle used to write to a register or, in 8-bit I/O mode, to a data transfer port. These accesses are entirely asynchronous, with the AT/LANTIC Controller responding when it decodes the correct address on SA0-9 and an  $\overline{\text{IOWR}}$ . If AEN is high the cycle will be ignored. CHRDY is deasserted if the AT/LANTIC Controller is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted. The data will always be taken from SD0-7.

## 6.0 Operation of AT/LANTIC Controller (Continued)

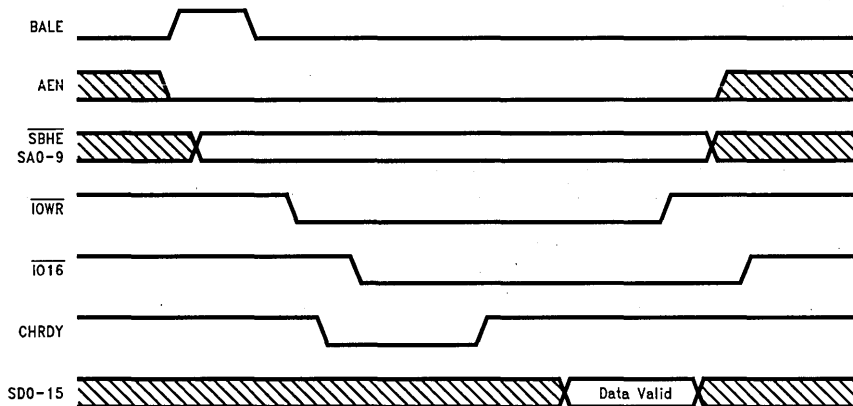
### 16-Bit I/O Mode Slave Read



TL/F/11498-41

This is the type of cycle used to read from a data transfer port in 16-bit I/O mode. These accesses are entirely asynchronous, with the AT/LANTIC Controller responding when it decodes the correct address on SA0-9 and an  $\overline{\text{IORD}}$ . If AEN is high the cycle will be ignored. CHRDY is deasserted if the AT/LANTIC Controller is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted.  $\overline{\text{IO16}}$  is generated, when an address within the AT/LANTIC Controller's data transfer port is decoded, to indicate to the system that this is a 16-bit transfer. If the  $\overline{\text{IO16CON}}$  bit in Configuration Register B is low then it will be a straight decode of the SA0-9 lines. If that bit is high the  $\overline{\text{IO16}}$  output will be generated after  $\overline{\text{IORD}}$  goes active.  $\overline{\text{SBHE}}$  must be low, to indicate that this is a 16-bit transfer, and the address should be even, SA0 low. The data will appear on SD0-15.

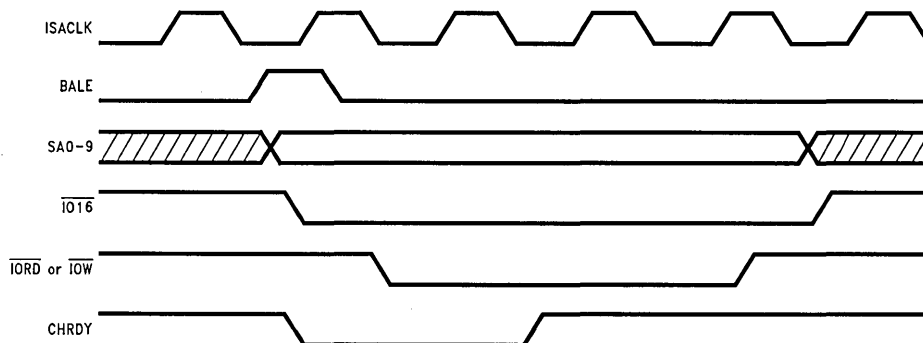
### 16-Bit I/O Mode Slave Write



TL/F/11498-42

This is the type of cycle used to write to a data transfer port in 16-bit I/O mode. These accesses are entirely asynchronous, with the AT/LANTIC Controller responding when it decodes the correct address on SA0-9 and an  $\overline{\text{IOWR}}$ . If AEN is high the cycle will be ignored. CHRDY is deasserted if the AT/LANTIC Controller is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted.  $\overline{\text{IO16}}$  is generated, when an address within the AT/LANTIC Controller's data transfer port is decoded, to indicate to the system that this is a 16-bit transfer. If the  $\overline{\text{IO16CON}}$  bit in Configuration Register B is low then it will be a straight decode of the SA0-9 lines. If that bit is high the  $\overline{\text{IO16}}$  output will be generated after  $\overline{\text{IOWR}}$  goes active.  $\overline{\text{SBHE}}$  must be low, to indicate that this is a 16-bit transfer, and the address should be even, SA0 low. The data will be taken from SD0-15.

## 6.0 Operation of AT/LANTIC Controller (Continued)



TL/F/11498-63

### 16-Bit I/O Cycle with CHRDY Fix

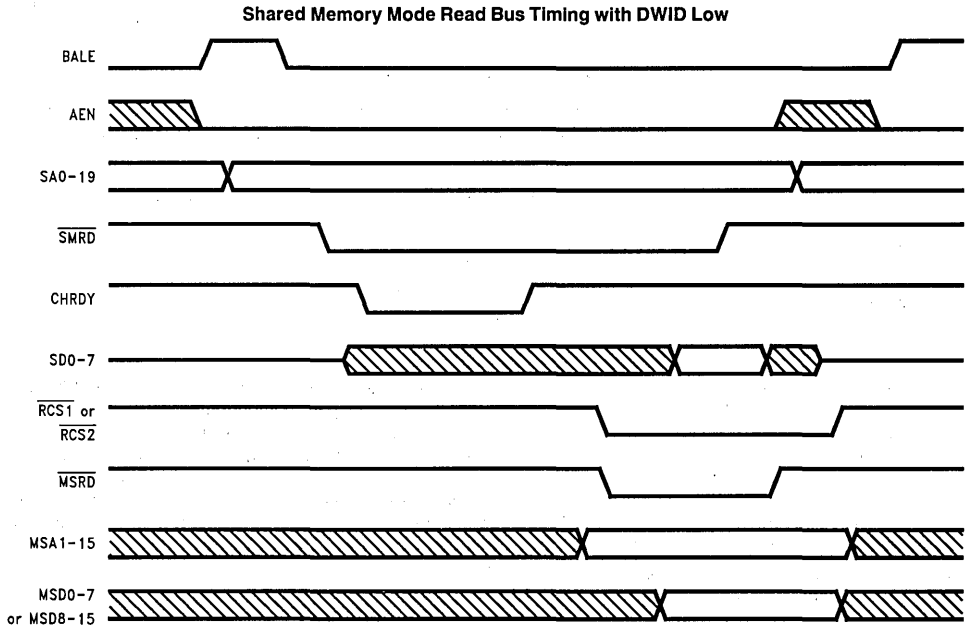
Some Chips and Technologies and VLSI Technologies PC-AT chip sets have timing requirements in 16-bit I/O cycles that cannot be achieved by the default AT/LANTIC cycle, described on the previous page. When that cycle is executed with these chip sets the system does not recognize the CHRDY signal and does not insert wait states. The system executes a standard cycle and deasserts IORD or IOW even if CHRDY is still deasserted. The AT/LANTIC recognizes if this situation has occurred, asserts CHRDY and sets a bus error bit in Configuration Register B to flag this error. Thus the user can test any new system to see if this error occurs and then take some remedial action. There are two ways of overcoming this problem, which are implemented by various board vendors. The AT/LANTIC supports both methods to allow the user to decide. Either fix can be selected by software, by writing to Configuration Register B.

The first fix is enabled by setting the IO16-bit of Configuration Register B. In normal operation any time a valid address exists on SA0-9 IO16 is generated. Delaying IO16 until after the IORD or IOW can cure the problem on non-compliant machines. The theory is that the system is fooled into thinking an 8-bit peripheral is responding, since IO16 is not generated for the valid address, and accepts 8-bit I/O cycle timings for CHRDY. It then rechecks IO16 after the IORD or IOW strobe and correctly determines it is a 16-bit peripheral. If a system did not recheck IO16 it would generate 2 8-bit cycles instead of 1 16-bit cycle. The AT/LANTIC would interpret each 8-bit access as a 16-bit transfer and decrement it's DMA byte count by 2. Eventually the system would attempt to access the data transfer port when the AT/LANTIC had finished transferring data and CHRDY would be deasserted indefinitely. To prevent misoperation, this fix should only be implemented on systems that require it.

The above figure shows the second fix to the problem with non-compliant machines. It is enabled by setting the CHRDY bit of Configuration Register B. This approach works on the theory that CHRDY deassertion is not fast enough and should be faster. In fact, it must be deasserted before the IORD or IOW strobe to operate correctly in some machines. All of the signals shown above are the same as a normal 16-bit I/O cycle, except CHRDY. BALE goes active and the address becomes valid after a falling edge of ISACK. This causes the AT/LANTIC to generate IO16 if the address decodes to the data transfer port. BALE goes inactive after the next rising edge of ISACK and IORD or IOW is asserted after the following falling edge. Normally CHRDY would be deasserted after the IORD or IOW strobe, if the AT/LANTIC was not ready. With this fix implemented CHRDY is deasserted as soon as the address becomes valid and BALE is active. If a memory cycle is in operation, instead of an I/O, CHRDY is asserted after the command strobe (MRD, MWR, SMRD or SMWR). If the address becomes invalid CHRDY is asserted. To prevent CHRDY being asserted for the half clock between BALE going inactive and IORD or IOW going active the AT/LANTIC holds CHRDY asserted as long as ISACK is high between these signals. If the delay between the falling edge of ISACK and the falling edge of IORD or IOW is too great, there may be a period where CHRDY is not held deasserted. This should not cause a problem. To prevent misoperation, this fix should only be implemented on systems that require it.

## 6.0 Operation of AT/LANTIC Controller (Continued)

### ISA Bus Shared Memory Access Timing

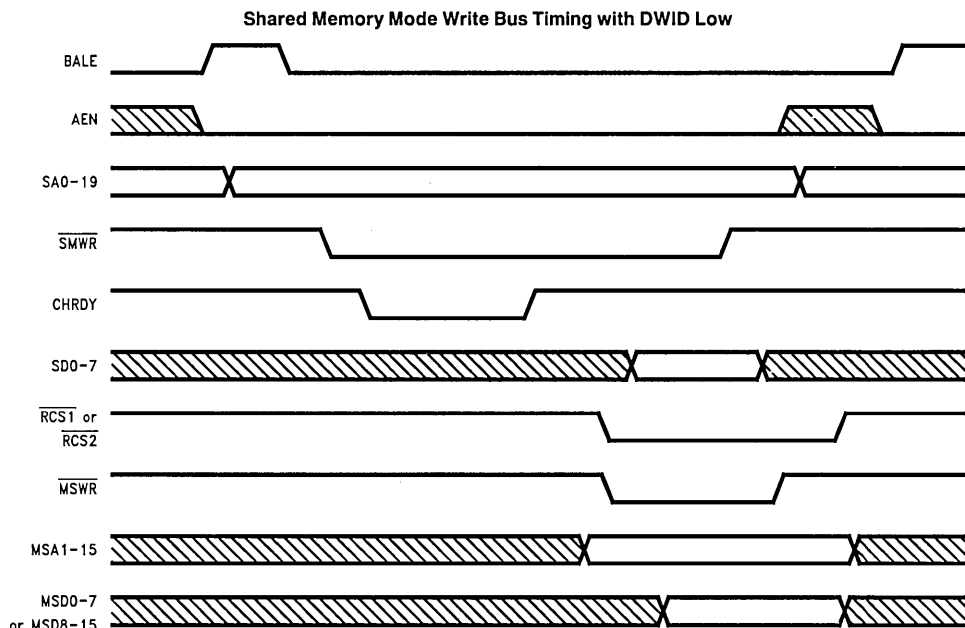


TL/F/11498-43

This is the type of cycle used to read from buffer RAM in shared memory mode when DWID is low. These accesses are entirely asynchronous, with the AT/LANTIC Controller responding when it decodes the address on SA0-19 matches Control Register 1 and an  $\overline{\text{SMRD}}$ . If AEN is high the cycle will be ignored. CHRDY is deasserted if the AT/LANTIC Controller is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted. If the memory width bit in Control Register 2 is low then 8 kbytes of RAM are accessible, so only  $\overline{\text{RCS1}}$  is used to strobe data and the data is always on MSD0-7. If this bit is high 16 kbytes of RAM are accessible, so both chip selects and byte lanes are used. If the memory address is even  $\overline{\text{RCS1}}$  and MSD0-7 are used, if odd  $\overline{\text{RCS2}}$  and MSD8-15 are used. System data is always output on SD0-7.



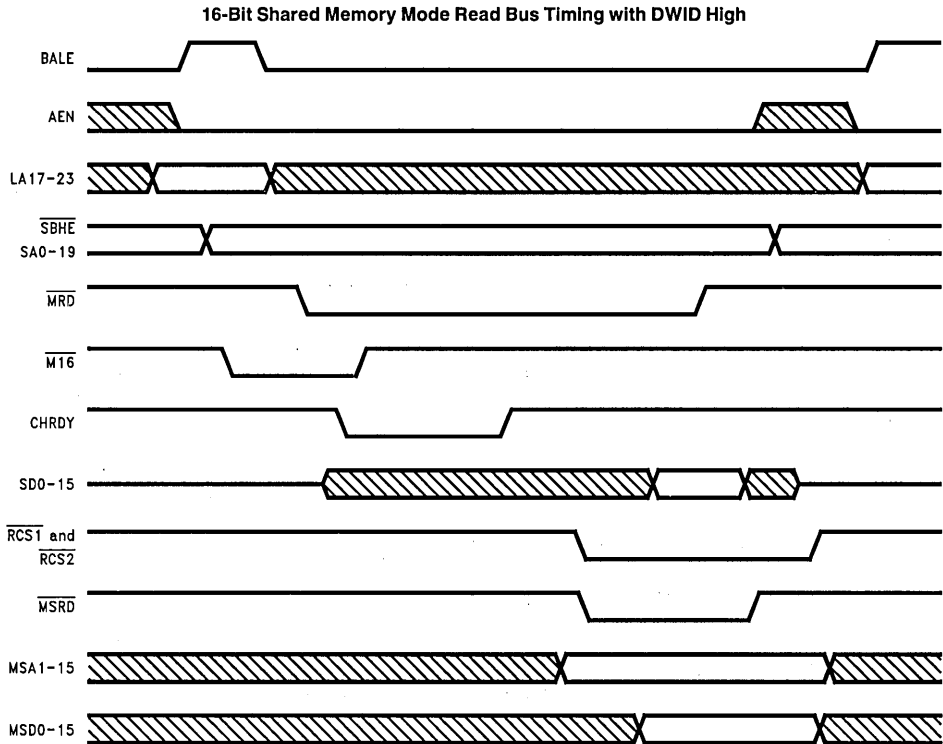
## 6.0 Operation of AT/LANTIC Controller (Continued)



TL/F/11498-44

This is the type of cycle used to write to buffer RAM in shared memory mode when DWID is low. These accesses are entirely asynchronous, with the AT/LANTIC Controller responding when it decodes the address on SA0-19 matches Control Register 1 and an  $\overline{SMWR}$ . If AEN is high the cycle will be ignored.  $\overline{CHRDY}$  is deasserted if the AT/LANTIC Controller is not ready to respond and asserted when ready. If it is ready immediately  $\overline{CHRDY}$  is not deasserted. If the memory width bit in Control Register 2 is low then 8 kbytes of RAM are accessible, so only  $\overline{RCS1}$  is used to strobe data and the data is always on MSD0-7. If this bit is high 16 kbytes of RAM are accessible, so both chip selects and byte lanes are used. If the memory address is even  $\overline{RCS1}$  and MSD0-7 are used, if odd  $\overline{RCS2}$  and MSD8-15 are used. System data is always taken from SD0-7.

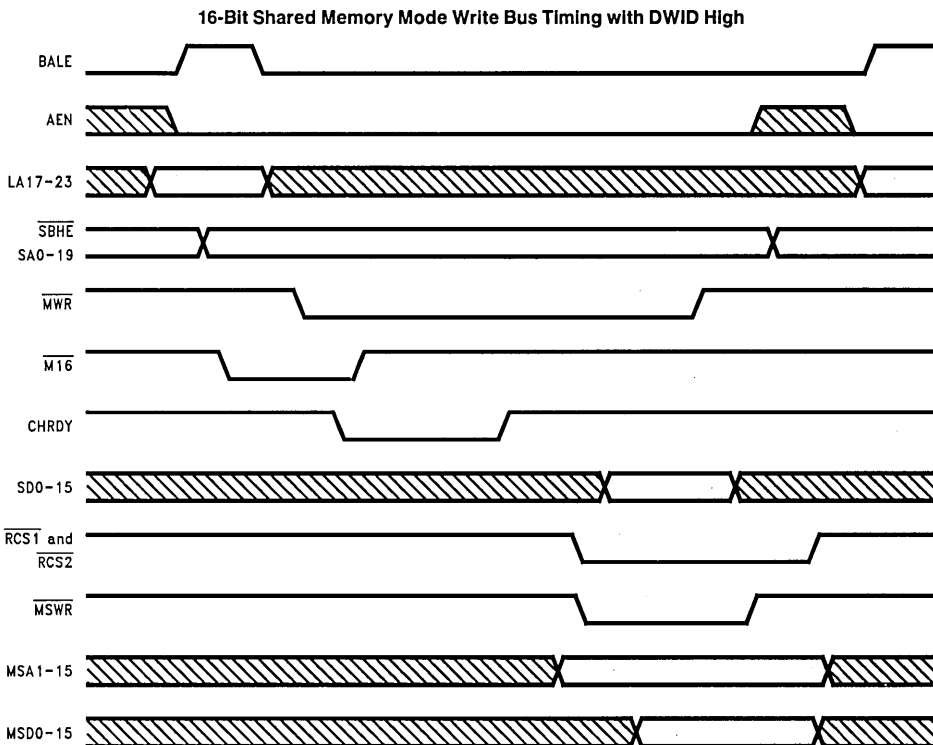
## 6.0 Operation of AT/LANTIC Controller (Continued)



TL/F/11498-45

This is the type of cycle used to read 16 bits from buffer RAM in shared memory mode when DWID is high. These accesses are entirely asynchronous, with the AT/LANTIC Controller responding when it decodes the correct address on LA17-23, SA0-19 and a MRD. If AEN is high the cycle will be ignored. CHRDY is deasserted if the AT/LANTIC Controller is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted. The 8/16-bit in Control Register 2 must be set to allow generation of M16. This will be generated whenever the LA17-23 lines match the corresponding values in Control Register 2. It will therefore be generated for a full 128 kbytes of address space, although the AT/LANTIC Controller will occupy less than that. It may be preferable to only set the 8/16-bit for the duration of a transfer from the buffer RAM. The AT/LANTIC Controller will also compare the address line programmed in Control Register 1 before allowing accesses to buffer RAM and therefore do a complete decode. The system indicates that this is a 16-bit transfer by asserting SBHE and accessing an even address, SA0 low. The full 16 bits of data bus are used on both system and memory support busses.

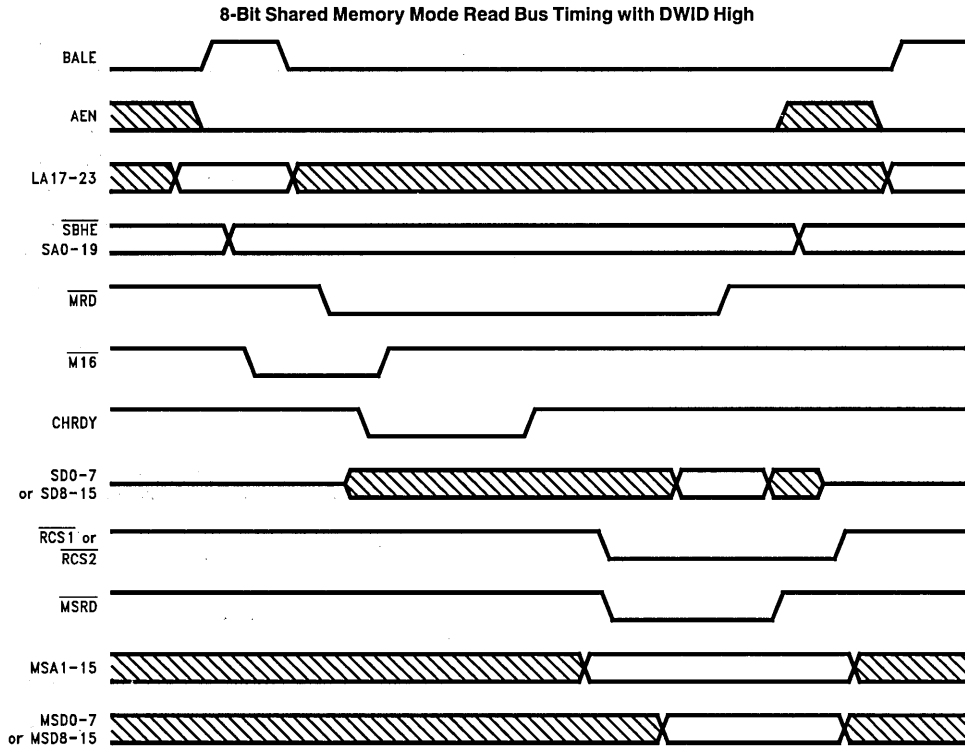
## 6.0 Operation of AT/LANTIC Controller (Continued)



TL/F/11498-46

This is the type of cycle used to write 16 bits to buffer RAM in shared memory mode when DWID is high. These accesses are entirely asynchronous, with the AT/LANTIC Controller responding when it decodes the correct address on LA17-23, SA0-19 and a MWR. If AEN is high the cycle will be ignored. CHRDY is deasserted if the AT/LANTIC Controller is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted. The 8/16-bit in Control Register 2 must be set to allow generation of M16. This will be generated whenever the LA17-23 lines match the corresponding values in Control Register 2. It will therefore be generated for a full 128 kbytes of address space, although the AT/LANTIC Controller will occupy less than that. It may be preferable to only set the 8/16-bit for the duration of a transfer to the buffer RAM. The AT/LANTIC Controller will also compare the address line programmed in Control Register 1 before allowing accesses to buffer RAM and therefore do a complete decode. The system indicates that this is a 16-bit transfer by asserting SBHE and accessing an even address, SA0 low. The full 16 bits of data bus are used on both system and memory support busses.

## 6.0 Operation of AT/LANTIC Controller (Continued)

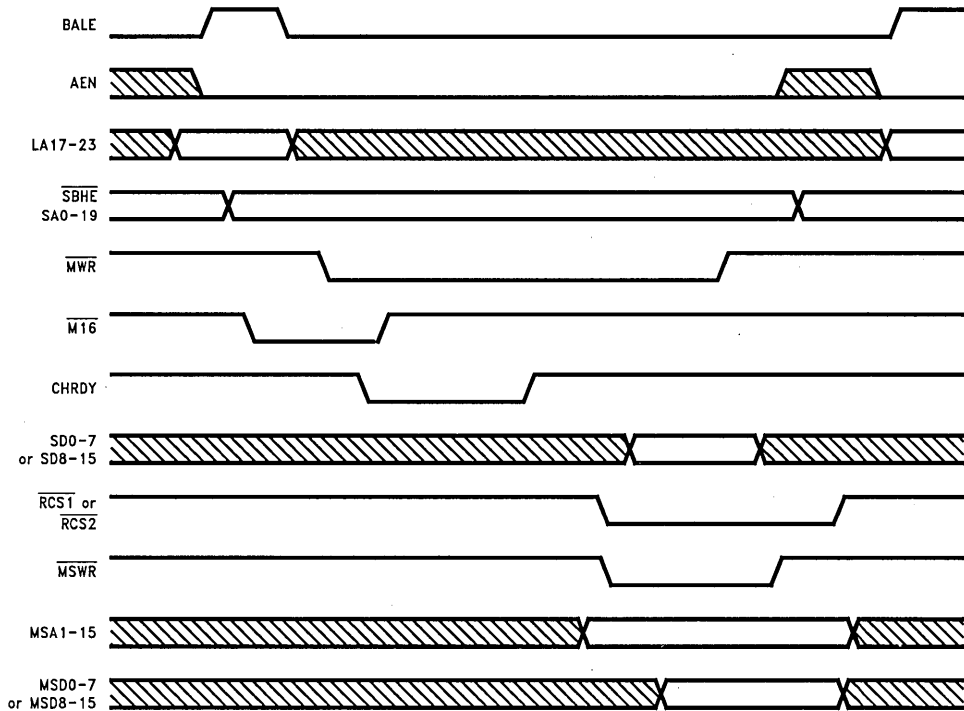


TL/F/11498-47

This is the type of cycle used to read 8 bits from buffer RAM in shared memory mode when DWID is high. These accesses are entirely asynchronous, with the AT/LANTIC Controller responding when it decodes the correct address on LA17-23, SA0-19 and a MRD. If AEN is high the cycle will be ignored. CHRDY is deasserted if the AT/LANTIC Controller is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted. The 8/16-bit in Control Register 2 must be set to allow generation of M16. This will be generated whenever the LA17-23 lines match the corresponding values in Control Register 2. It will therefore be generated for a full 128 kbytes of address space, although the AT/LANTIC Controller will occupy less than that. It may be preferable to only set the 8/16-bit for the duration of a transfer from the buffer RAM. The AT/LANTIC Controller will also compare the address line programmed in Control Register 1 before allowing accesses to buffer RAM and therefore do a complete decode. The system indicates that this is an 8-bit transfer by not asserting SBHE for an even address, SA0 low, or by accessing an odd address, SA0 high. If the 8/16-bit is low the AT/LANTIC Controller will only drive data onto SD0-7. Even addresses will use RCS1 and MSD0-7, odd addresses will use RCS2 and MSD8-15. If 8/16-bit is high the AT/LANTIC Controller can drive either SD0-7 or SD8-15. Even addresses are fetched using RCS1 and MSD0-7 and driven onto SD0-7. Odd addresses are fetched using RCS2 and MSD8-15 and driven onto SD8-15.

## 6.0 Operation of AT/LANTIC Controller (Continued)

8-Bit Shared Memory Mode Write Bus Timing with DWID High

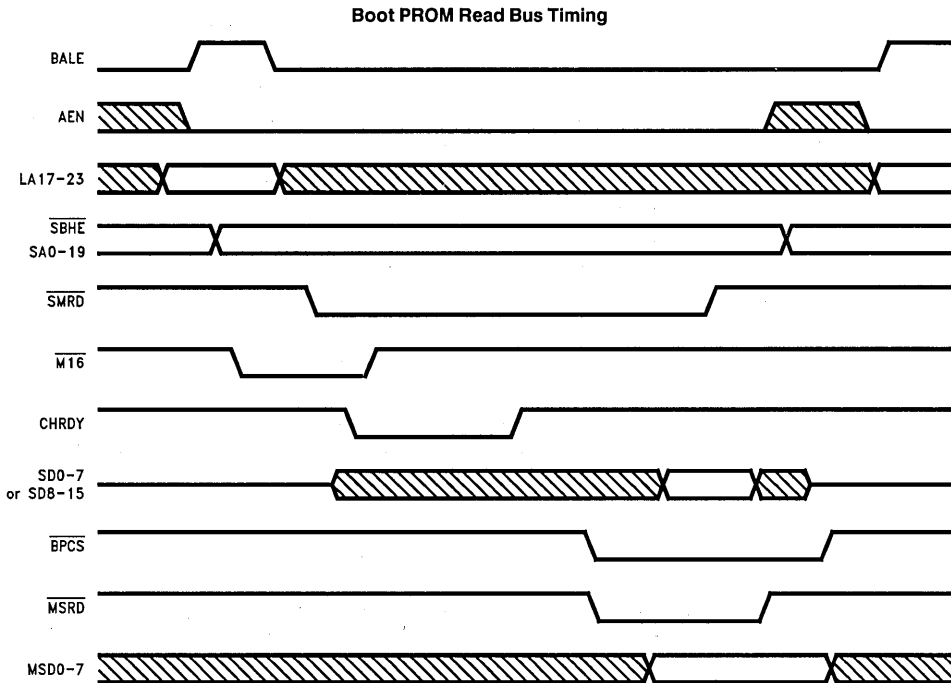


TL/F/11498-48

This is the type of cycle used to write 8 bits to buffer RAM in shared memory mode when DWID is high. These accesses are entirely asynchronous, with the AT/LANTIC Controller responding when it decodes the correct address on LA17-23, SA0-19 and a  $\overline{IORD}$ . If AEN is high the cycle will be ignored. CHRDY is deasserted if the AT/LANTIC Controller is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted. The 8/16-bit in Control Register 2 must be set to allow generation of M16. This will be generated whenever the LA17-23 lines match the corresponding values in Control Register 2. It will therefore be generated for a full 128 kbytes of address space, although the AT/LANTIC Controller will occupy less than that. It may be preferable to only set the 8/16-bit for the duration of a transfer to the buffer RAM. The AT/LANTIC Controller will also compare the address line programmed in Control Register 1 before allowing accesses to buffer RAM and therefore do a complete decode. The system indicates that this is an 8-bit transfer by not asserting SBHE for an even address, SA0 low, or by accessing an odd address, SA0 high. If the 8/16-bit is low the AT/LANTIC Controller will only read data from SD0-7. Even addresses will use RCS1 and MSD0-7, odd addresses will use RCS2 and MSD8-15. If 8/16-bit is high the AT/LANTIC Controller can read from either SD0-7 or SD8-15. Even addresses are read from SD0-7 and written to RAM using RCS1 and MSD0-7. Odd addresses are read from SD8-15 and written to RAM using RCS2 and MSD8-15.

## 6.0 Operation of AT/LANTIC Controller (Continued)

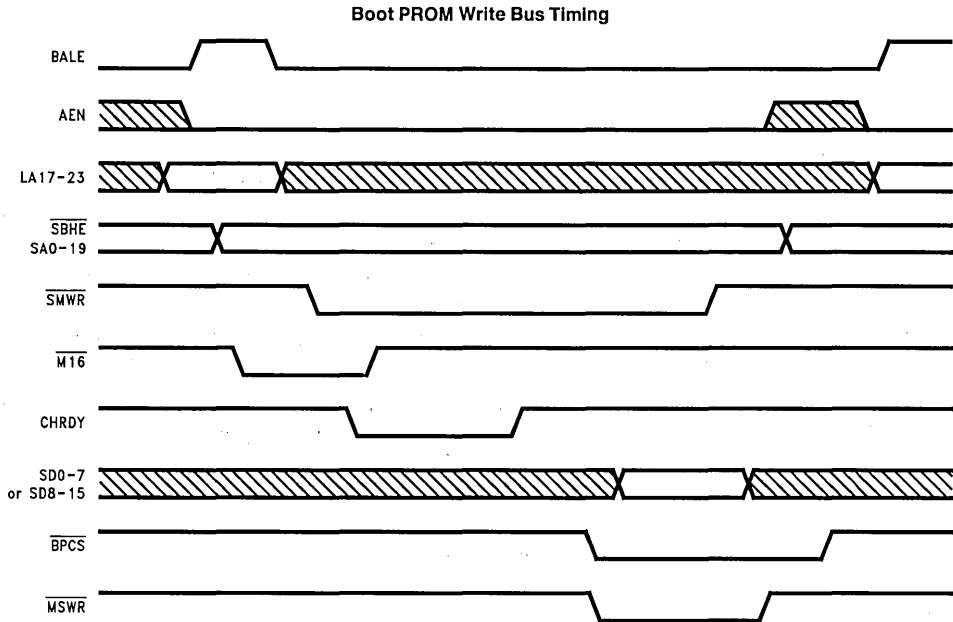
### ISA Bus Boot PROM Access Timing



TL/F/11498-49

This is the type of cycle used to read the boot PROM. These accesses are entirely asynchronous, with the AT/LANTIC Controller responding when it decodes the correct address on SA0-19 and a  $\overline{\text{SMRD}}$ . If AEN is high the cycle will be ignored. CHRDY is deasserted if the AT/LANTIC Controller is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted.  $\overline{\text{M16}}$  is only generated if the AT/LANTIC Controller is 1) in shared memory mode AND 2) DWID is high AND 3) 8/16-bit in Control Register 2 is high AND 4) the LA17-23 lines match the corresponding values in Control Register 2. The data will normally be driven onto SD0-7. However, if  $\overline{\text{M16}}$  is generated and the access is to an odd address the data will be driven onto SD8-15. The data will always be taken from MSD0-7.

## 6.0 Operation of AT/LANTIC Controller (Continued)

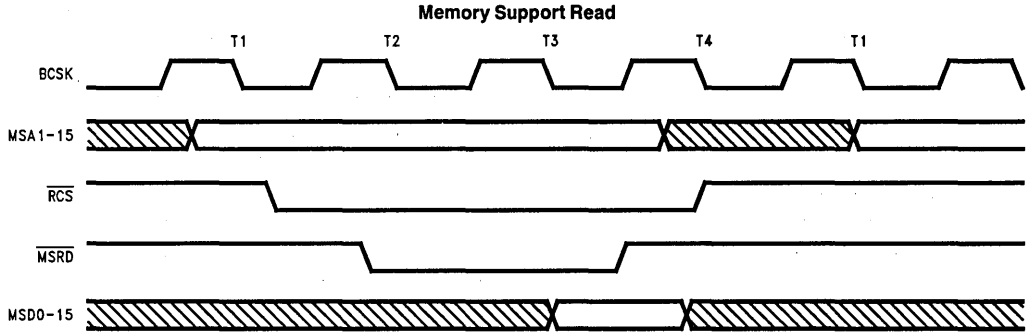


TL/F/11498-50

This is the type of cycle used to write to the boot PROM. These accesses are entirely asynchronous, with the AT/LANTIC Controller responding when it decodes the correct address on SA0-19 and a  $\overline{SMWR}$ . If AEN is high the cycle will be ignored. CHRDY is deasserted if the AT/LANTIC Controller is not ready to respond and asserted when ready. If it is ready immediately CHRDY is not deasserted.  $\overline{M16}$  is only generated if the AT/LANTIC Controller is 1) in shared memory mode AND 2) DWID is high AND 3) 8/16-bit in Control Register 2 is high AND 4) the LA17-23 lines match the corresponding values in Control Register 2. The data will normally be taken from SD0-7. However, if  $\overline{M16}$  is generated and the access is to an odd address the data will be taken from SD8-15. The data will always be driven onto MSD0-7. The BPWR bit of Configuration Register B must be high to allow write cycles to the boot PROM.

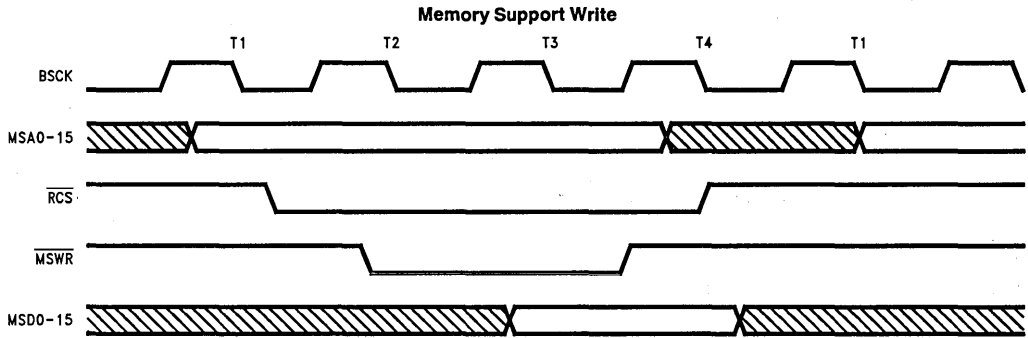
## 6.0 Operation of AT/LANTIC Controller (Continued)

### RAM Access Timing



TL/F/11498-51

This is a memory read cycle executed by the AT/LANTIC Controller's internal DMA. This is used to either load the data transfer port, during a Remote Read in I/O mode, or to load the FIFO, for a transmission on the network, in both modes. This transfer is synchronized to BCLK, which can be either driven from the 20 MHz input on X1 or by the BCLK input. This is selected by the CLKSEL bit in Configuration Register C. If there is 8 kbytes of RAM only  $\overline{RCS1}$  is used, if 16 kbytes are available  $\overline{RCS1}$  and  $\overline{RCS2}$  are used.



TL/F/11498-52

This is a memory write cycle executed by the AT/LANTIC Controller's internal DMA. This is used to either write from the data transfer port, during a Remote Write in I/O mode, or to empty the FIFO, during a reception from the network, in both modes. This transfer is synchronized to BCLK, which can be either driven from the 20 MHz input on X1 or by the BCLK input. This is selected by the CLKSEL bit in Configuration Register C. If there is 8 kbytes of RAM only  $\overline{RCS1}$  is used, if 16 kbytes are available  $\overline{RCS1}$  and  $\overline{RCS2}$  are used.



## 7.0 Preliminary Electrical Characteristics

## Operating Conditions

	Min	Max	Units
Supply Voltage ( $V_{CC}$ )	4.75	5.25	V
Operating Temperature ( $T_A$ )	0	+70	°C
ESD Tolerance:	1.25		kV
$C_{ZAP} = 100 \text{ pF}$ , $R_{ZAP} = 1.5 \text{ k}\Omega$			

## Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage ( $V_{CC}$ )	-0.5V to +7.0V
Storage Temperature ( $T_{STG}$ )	-65°C to +150°C
Package Power Dissipation ( $P_D$ )	800 mW
Lead Temperature ( $T_L$ ) (Soldering, 10 seconds)	260°C

## Preliminary DC Specifications

Symbol	Description	Conditions	Min	Max	Units
<b>SUPPLY CURRENT</b>					
$I_{CC}$	Average Active (Transmitting/Receiving) Supply Current	$X1 = 20 \text{ MHz Clock}$ $V_{IN} = \text{Switching}$		100	mA
$I_{CCIDLE}$	Average Idle Supply Current	$X1 = 20 \text{ MHz Clock}$ $V_{IN} = V_{CC} \text{ or GND}$		80	mA
$I_{CCLP}$	Low Power Supply Current	$X1 = \text{Undriven}$ $V_{IN} = V_{CC} = \text{Undriven}$		35	$\mu\text{A}$
<b>TTL INPUTS</b>					
$V_{IL}$	Maximum Low Level Input Voltage			0.8	V
$V_{IH}$	Minimum High Level Input Voltage		2.0		V
$I_{IN}$	Input Current	$V_I = V_{CC} \text{ or GND}$	-1.0	+1.0	$\mu\text{A}$
<b>3SH TRI-STATE HIGH DRIVE I/O</b>					
$V_{OH}$	Minimum High Level Output Voltage	$I_{OH} = -3 \text{ mA}$	2.4		V
$V_{OL}$	Maximum Low Level Output Voltage	$I_{OL} = 24 \text{ mA}$		0.5	V
$V_{IL}$	Maximum Low Level Input Voltage			0.8	V
$V_{IH}$	Minimum High Level Input Voltage		2.0		V
$I_{IN}$	Input Current	$V_I = V_{CC} \text{ or GND}$	-1.0	+1.0	$\mu\text{A}$
$I_{OZ}$	Maximum TRI-STATE Output Leakage Current	$V_{OUT} = V_{CC} \text{ or GND}$	-10.0	+10.0	$\mu\text{A}$
<b>MOS INPUTS, OUTPUTS AND I/O</b>					
$V_{OH}$	Minimum High Level Output Voltage	$I_{OH} = -20 \mu\text{A}$	$V_{CC} - 0.1$		V
$V_{OL}$	Maximum Low Level Output Voltage	$I_{OL} = 20 \mu\text{A}$		0.1	V
$V_{IL}$	Maximum Low Level Input Voltage			0.8	V
$V_{IH}$	Minimum High Level Input Voltage		2.0		V
$V_{ILD}$	Maximum Low Level Input Voltage DWID			1.0	V
$V_{IHD}$	Minimum High Level Input Voltage DWID		4.0		V
$I_{IN}$	Input Current	$V_I = V_{CC} \text{ or GND}$	-1.0	+1.0	$\mu\text{A}$
$I_{IND}$	Input Current TEST, DWID Pull Down Resistor	$V_I = V_{CC}$		2000	$\mu\text{A}$
$I_{IN2}$	Input Current TEST, MSD0-7	$V_I = V_{CC} \text{ or GND}$		2000	$\mu\text{A}$
$I_{IN3}$	Input Current TEST, MSD8-15, MSA1-8	$V_I = V_{CC} \text{ or GND}$ RESET = Active		2000	$\mu\text{A}$
$I_{OZ}$	Maximum TRI-STATE Output Leakage Current	$V_{OUT} = V_{CC} \text{ or GND}$	-10.0	+10.0	$\mu\text{A}$

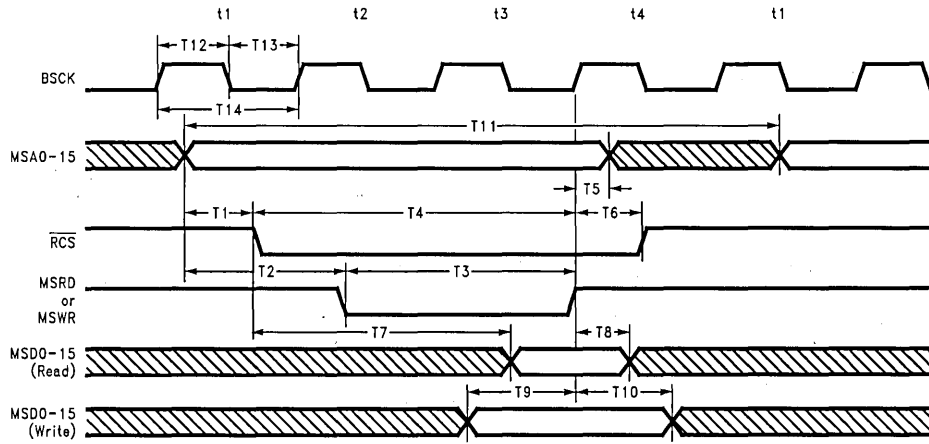
**7.0 Preliminary Electrical Characteristics** (Continued)**Preliminary DC Specifications** (Continued)

Symbol	Description	Conditions	Min	Max	Units
<b>OCH OPEN COLLECTOR HIGH DRIVE OUTPUT</b>					
V <sub>OL</sub>	Maximum Low Level Output Voltage	I <sub>OL</sub> = 24 mA		0.5	V
<b>LED DRIVER OUTPUT</b>					
V <sub>OL</sub>	Maximum Low Level Output Voltage	I <sub>OL</sub> = 16 mA		0.5	V
<b>THIN DRIVER OUTPUT</b>					
V <sub>OH</sub>	Minimum High Level Output Voltage	I <sub>OH</sub> = -8 mA	2.4		V
V <sub>OL</sub>	Maximum Low Level Output Voltage	I <sub>OL</sub> = 2 mA		0.5	V
<b>OSCILLATOR PINS (X1 AND X2)</b>					
V <sub>IH</sub>	X1 Input High Voltage	X1 is Connected to an Oscillator	2.0		V
V <sub>IL</sub>	X1 Input Low Voltage	X1 is Connected to an Oscillator		0.8	V
I <sub>OSC</sub>	X1 Input Current	X1 is Connected to an Oscillator V <sub>IN</sub> = V <sub>CC</sub> or GND		1	mA
<b>AUI</b>					
V <sub>OD</sub>	Differential Output Voltage (TX ±)	78Ω Termination and 270Ω from Each to GND (Note 1)	±550	±1200	mV
V <sub>OB</sub>	Differential Idle Output Voltage Imbalance (TX ±)	78Ω Termination and 270Ω from Each to GND (Note 1)	Typical: 40 mV		
V <sub>U</sub>	Undershoot Voltage (TX ±)	78Ω Termination and 270Ω from Each to GND (Note 1)	Typical: 80 mV		
V <sub>DS</sub>	Diff. Squelch Threshold (RX ±, CD ±)		-175	-300	mV
V <sub>CM</sub>	Diff. Input Common Mode Voltage (RX ±, CD ±)	(Note 1)	0	5.25	V
<b>TPI</b>					
R <sub>TOL</sub>	TXO <sub>d</sub> ±, TXO ± Low Level Output Resistance	I <sub>OL</sub> = 25 mA		15	Ω
R <sub>TOH</sub>	TXO <sub>d</sub> ±, TXO ± High Level Output Resistance	I <sub>OH</sub> = -25 mA		15	Ω
V <sub>SRON1</sub>	Receive Threshold Turn-On Voltage 10BASE-T Mode		±300	±585	mV
V <sub>SRON2</sub>	Receive Threshold Turn-On Voltage Reduced Threshold		±175	±300	mV
V <sub>SROFF</sub>	Receive Threshold Turn-Off Voltage	(Note 1)	±175	±300	mV
V <sub>DIFF</sub>	Differential Mode Input Voltage Range	V <sub>CC</sub> = 5.0 V (Note 1)	-3.1	+3.1	V

**Note 1:** These parameters are not guaranteed by production testing.

## 8.0 Preliminary Switching Characteristics

Memory Support Bus Accesses (for I/O port or FIFO transfers)



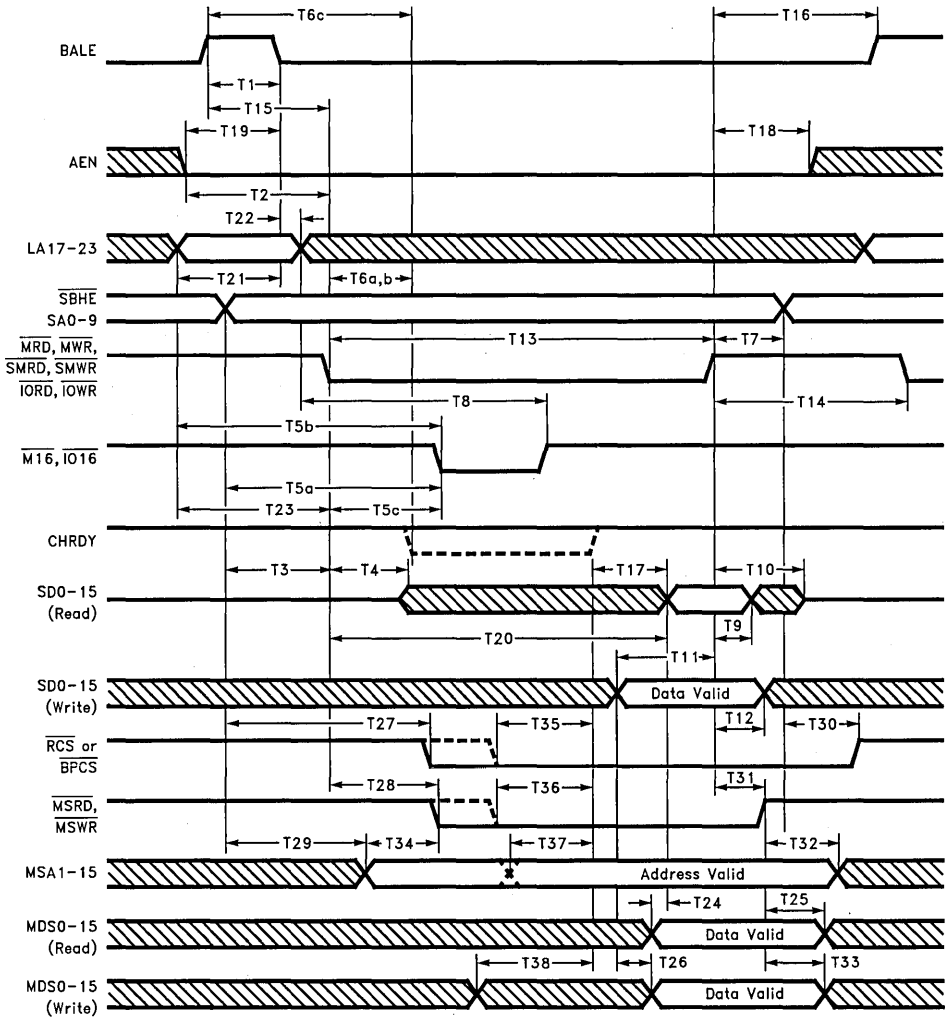
TL/F/11498-53

Symbol	Description	8-Bit Transfers		16-Bit Transfers		Units
		Min	Max	Min	Max	
T1	MSA1 – 15 Valid before $\overline{RCS}$ Asserted (Note 1)		30	30		ns
T2	MSA1 – 15 Valid before $\overline{MSRD}$ – WR Asserted	20		20		ns
T3	$\overline{MSRD}$ – WR Width	70		70		ns
T4	$\overline{RCS}$ and MSA1 – 15 Valid to $\overline{MSWR}$ Deasserted (Note 1)	105		105		ns
T5	MSA1 – 15 Valid after $\overline{MSRD}$ – WR Deasserted	10		10		ns
T6	$\overline{RCS}$ Held after $\overline{MSRD}$ – WR Deasserted (Note 1)	10		10		ns
T7	$\overline{RCS}$ and MSA1 – 15 Valid to $\overline{MSD0}$ – 15 Valid (Note 1)		100		100	ns
T8	Read Data Hold from $\overline{MSRD}$ Deasserted	0		0		ns
T9	Write Data Set-Up to $\overline{MSWR}$ Deasserted	50		50		ns
T10	Write Data Held from $\overline{MSWR}$ Deasserted	10		10		ns
T11	Time Between Transfers	4		4		bcyc
T12	Minimum Bus Clock High Time (bch)	20		20		ns
T13	Minimum Bus Clock Low Time (bcl)	20		20		ns
T14	Minimum Bus Clock Cycle Time (bcyc)	50		50		ns

Note 1: In 8-bit mode  $\overline{RCS}$  refers to  $\overline{RCST}$  only. In 16-bit mode  $\overline{RCS}$  refers to both  $\overline{RCST}$  and  $\overline{RCS2}$ .

# 8.0 Preliminary Switching Characteristics (Continued)

## ISA Slave Accesses



TL/F/11498-54

## 8.0 Preliminary Switching Characteristics (Continued)

### ISA Slave Accesses

Symbol	Description	8-Bit Transfers		16-Bit Transfers		Units
		Min	Max	Min	Max	
T1	BALE Width	20		20		ns
T2	AEN Valid before Command Strobe Active	60		60		ns
T3a	$\overline{SBHE}$ and SA0–9 Valid before $\overline{IORD}$ , $\overline{IOWR}$ Asserted	40		20		ns
T3b	SA0–9 Valid before $\overline{MRD}$ , $\overline{MWR}$ Asserted	32		32		ns
T4a	$\overline{IORD}$ , $\overline{MRD}$ Asserted to SD0–15 Driven (Note 3)	0		0		ns
T5a	$\overline{SBHE}$ and SA0–9 Valid before $\overline{IOT16}$ Valid (Notes 1, 9)				60	ns
T5b	LA17–23 Valid to $\overline{MT16}$ Valid (Note 1)				55	ns
T5c	$\overline{SBHE}$ and SA0–9 Valid and $\overline{IORD}$ or $\overline{IOWR}$ Active before $\overline{IOT16}$ Valid (Notes 1, 10)				50	ns
T6a	$\overline{IORD}$ , $\overline{IOWR}$ Asserted to CHRDY Deasserted (Notes 2, 5)		100		50	ns
T6b	$\overline{MRD}$ , $\overline{MWR}$ Asserted to CHRDY Deasserted (Note 2)		45		45	ns
T6c	BALE Asserted and SA0–9 Valid to CHRDY Deasserted (Notes 2, 4)		60		60	ns
T7	$\overline{IORD}$ , $\overline{IOWR}$ Deasserted before $\overline{SBHE}$ and SA0–9 Invalid	15		15		ns
T8a	$\overline{SBHE}$ and SA0–9 Invalid to $\overline{IOT16}$ Invalid	0		0		ns
T8b	LA17–23 Invalid to $\overline{MT16}$ Invalid (Note 1)			0		ns
T9	$\overline{IORD}$ , $\overline{MRD}$ Deasserted to SD0–15 Read Data Invalid (Note 3)	0		0		ns
T10	$\overline{IORD}$ , $\overline{MRD}$ Deasserted to SD0–15 Floating (Note 3)		45		45	ns
T11a	D0–15 Write Data Valid to $\overline{IOWR}$ Deasserted (Note 3)	60		20		ns
T12	$\overline{IOWR}$ , $\overline{MWR}$ Deasserted to SD0–15 Write Data Invalid (Note 3)	20		20		ns
T13a	$\overline{IORD}$ , $\overline{IOWR}$ Active Width (Note 8)	300		140		ns
T14a	$\overline{IORD}$ , $\overline{IOWR}$ Inactive Width	85		85		ns
T14b	$\overline{SMRD}$ , $\overline{SMWR}$ , $\overline{MRD}$ , $\overline{MWR}$ Inactive Width					ns
T15	BALE Asserted before $\overline{MRD}$ , $\overline{MWR}$ Asserted			25		ns
T16	$\overline{MRD}$ , $\overline{MWR}$ Deasserted before Next BALE Asserted			20		ns
T17	CHRDY Asserted to SD0–15 I/O Read Data Valid (Notes 2, 3, 6)		60		60	ns
T18	$\overline{IORD}$ , $\overline{IOWR}$ Deasserted before AEN Invalid	25		25		ns
T19	AEN Valid before BALE Deasserted	50		50		ns
T20	$\overline{IORD}$ Asserted to SD0–15 Read Data Valid (Notes 3 and 7)		150		90	ns
T21	LA17–23 Valid before BALE Deasserted			40		ns

## 8.0 Preliminary Switching Characteristics (Continued)

### ISA Slave Accesses

Symbol	Description	8-Bit Transfers		16-Bit Transfers		Units
		Min	Max	Min	Max	
T22	BALE Deasserted before LA17–23 Invalid			0		ns
T23	LA17–23 Valid before $\overline{\text{MRD}}$ , $\overline{\text{MWR}}$ Asserted			40		ns
T24	Read Data Valid on MSD0–15 to Valid on SD0–15		70		70	ns
T25	$\overline{\text{MSRD}}$ Deasserted to MSD0–15 Read Data Invalid (Note 3)	0		0		ns
T26	Write Data Valid on SD0–15 to Valid on MSD0–15		65		65	ns
T27	SA0–19 Valid to $\overline{\text{RCS}}$ or $\overline{\text{BPCS}}$ Asserted (Note 11)		55		55	ns
T28a	$\overline{\text{MRD}}$ Asserted to $\overline{\text{MSRD}}$ Asserted		60		60	ns
T28b	$\overline{\text{MWR}}$ Asserted to $\overline{\text{MSWR}}$ Asserted		120		120	ns
T29	SA0–19 Valid to MSA1–15 Valid		60		60	ns
T30	SA0–19 Invalid to $\overline{\text{RCS}}$ or $\overline{\text{BPCS}}$ Deasserted (Note 11)	0		0		ns
T31	$\overline{\text{MRD}}$ , $\overline{\text{MSRD}}$ Deasserted to $\overline{\text{MSRD}}$ Deasserted	0		0		ns
T32	$\overline{\text{MSWR}}$ Deasserted to MA1–15 Invalid	10		10		ns
T33	$\overline{\text{MSWR}}$ Deasserted to MSD0–15 Invalid (Note 3)	0		0		ns
T34	MSA1–15 valid before $\overline{\text{MSWR}}$ Asserted	20		20		ns
T35a	$\overline{\text{RCS}}$ Asserted to CHRDY Asserted (Note 11, 12)	80		80		ns
T35b	$\overline{\text{BPCS}}$ Asserted to CHRDY Asserted (Note 13)	175		175		ns
T36a	$\overline{\text{MSRD}}$ , $\overline{\text{MSWR}}$ Asserted to CHRDY Asserted (Note 11)	15		15		ns
T36b	$\overline{\text{MSRD}}$ , $\overline{\text{MSWR}}$ Asserted to CHRDY Asserted (Note 13)	150		150		ns
T37	MSA1–15 Valid to CHRDY Asserted (Note 11)	75		75		ns
T38a	Driving Data from SD0–15 on to MSD0–15 to CHRDY Asserted (Note 11)	60		60		ns
T38b	Driving Data from SD0–15 on to MSD0–15 to CHRDY Asserted (Note 13)	260		260		ns

**Note 1:**  $\overline{\text{MT6}}$ ,  $\overline{\text{IOT6}}$  are only asserted for 16-bit transfers.

**Note 2:** CHRDY is only deasserted if the NIC core can not service the access immediately. It is held deasserted until the NIC core is ready, causing the system to insert wait states.

**Note 3:** On 8-bit transfers only 8 bits of MSD0–15 and D0–7 are driven.

**Note 4:** This is the early CHRDY timing, required by some machines, where CHRDY is referenced to BALE. In this mode of operation, under certain circumstances, CHRDY will be asserted for cycles which are not for this device, i.e. memory cycles or I/O cycles where SA0–9 match our address before reaching their valid state. In such a case the time to assert CHRDY, from  $\overline{\text{MRD}}$ ,  $\overline{\text{MWR}}$  or SA0–9 invalid, will be the same as the deassertion time specified.

**Note 5:** This is the standard CHRDY timing where CHRDY is asserted after  $\overline{\text{IORD}}$  or  $\overline{\text{IOWR}}$ .

**Note 6:** Read data valid is referenced to CHRDY when wait states have been inserted.

**Note 7:** If no wait states are inserted read data valid can be measured from  $\overline{\text{IORD}}$ .

**Note 8:** This is a minimum timing with no additional wait states.

**Note 9:** This is the standard  $\overline{\text{IOT6}}$  timing where  $\overline{\text{IOT6}}$  is asserted after a valid address decode.

**Note 10:** This is the late  $\overline{\text{IOT6}}$  timing, required by some machines, where  $\overline{\text{IOT6}}$  is asserted after a valid address decode and  $\overline{\text{IORD}}$  or  $\overline{\text{IOWR}}$  going active.

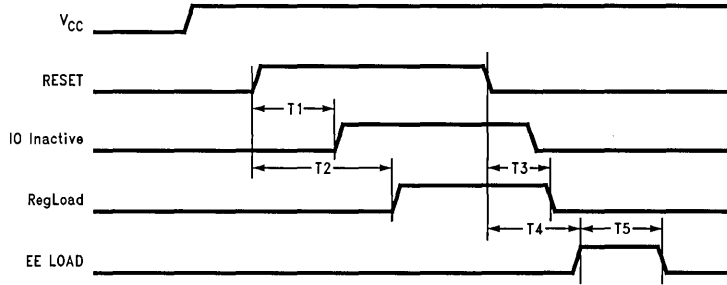
**Note 11:** This is a timing for a RAM access.

**Note 12:** RCS refers to  $\overline{\text{RCS1}}$  and  $\overline{\text{RCS2}}$ . Depending on the mode of operation either or both can be asserted. See the Functional Bus Timing section for a further explanation.

**Note 13:** This is a timing for a Boot PROM access.

## 8.0 Preliminary Switching Characteristics (Continued)

### RESET Timing



TL/F/11498-55

Symbol	Description	Min	Max	Units
T1	RESET Asserted Until IO Inactive Asserted (Note 1)	400		ns
T2	RESET Asserted Until RegLoad State Entered (Note 2)	415		μs
T3	RESET Deasserted Until RegLoad Deasserted (Note 3)	100		ns
T4	RESET Deasserted Until EELoad State Entered (Note 4)	0		μs
TS	EELoad Width (Note 4)		320	μs

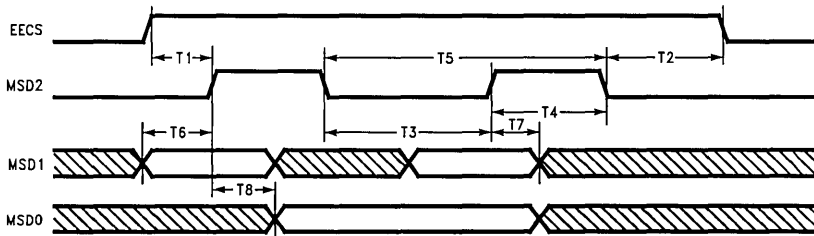
**Note 1:** I/O inactive is not an external signal. It is used here to indicate the length of time RESET must be active before the AT/LANTIC Controller recognizes it, begins to drive the ISA outputs to their inactive state and ignores ISA inputs except RESET.

**Note 2:** RegLoad is not an external signal. It is used here to indicate the length of time RESET must be active before the AT/LANTIC Controller begins configuring. When IOinactive goes active the internal pull-down resistors on the memory support buses are enabled.

**Note 3:** If RegLoad is high the values on the memory support buses are latched into the configuration registers when RESET is deasserted. The pull-down resistors on this bus are enabled until RegLoad is deasserted.

**Note 4:** EELoad is not an external signal, it is used here to indicate when the EEPROM store is loading.

### Serial EEPROM Timing



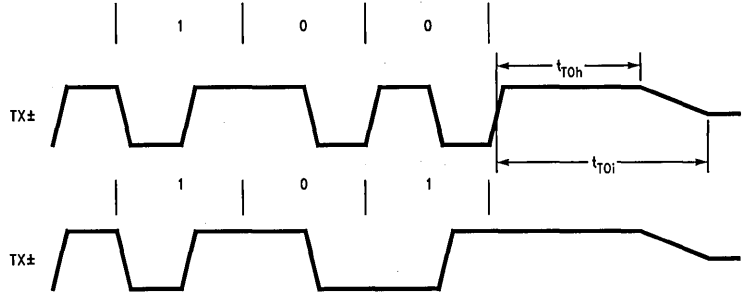
TL/F/11498-56

Symbol	Description	Min	Max	Units
T1	EECS Setup to SK	150		ns
T2	EECS Hold after SK	250		ns
T3	MSD2 Low Time	450		ns
T4	MSD2 High Time	450		ns
T5	MSD2 Clock Period (Note 1)	1		μs
T6	Data In Setup to MSD2 High	100		ns
T7	Data In Hold from MSD2 High	100		ns
T8	Data Out Valid from MSD2 High		500	ns

**Note 1:** Derived from Crystal Oscillator Tolerance = ±0.01%.

### 8.0 Preliminary Switching Characteristics (Continued)

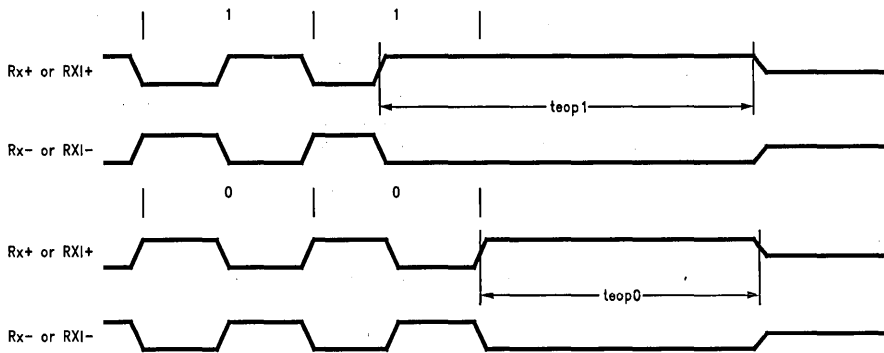
#### AUI Transmit Timing (End of Packet)



TL/F/11498-57

Symbol	Description	Min	Max	Units
$t_{TOH}$	Transmit Output High before Idle	200		ns
$t_{TOI}$	Transmit Output Idle Time	8000		ns

#### AUI/TPI Receive Timing (End of Packet)



TL/F/11498-58

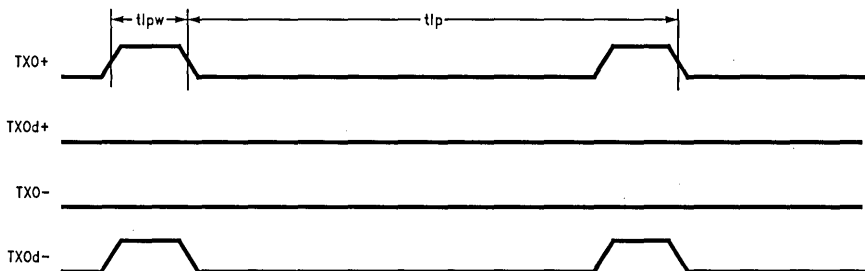
Symbol	Description	Min	Max	Units
$t_{eop1}$	Receive End of Packet Hold Time after Logic "1" (Note 1)	225		ns
$t_{eop0}$	Receive End of Packet Hold Time after Logic "0" (Note 1)	225		ns

**Note 1:** This parameter is guaranteed by design and is not tested.



## 8.0 Preliminary Switching Characteristics (Continued)

### Link Pulse Timing

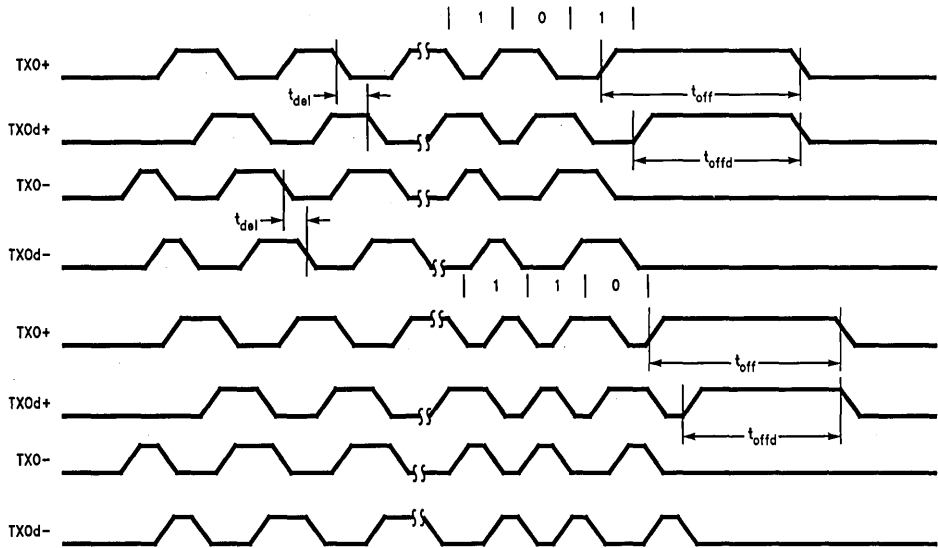


TL/F/11498-59

Symbol	Description	Min	Max	Units
$t_{lp}$	Time between Link Output Pulses	8	24	ns
$t_{lpw}$	Link Integrity Output Pulse Width	80	130	ns

## 8.0 Preliminary Switching Characteristics (Continued)

### TPI Transmit Timing (End of Packet)



TL/F/11498-60

Symbol	Description	Min	Max	Units
$t_{del}$	Pre-Emphasis Output Delay (TXO± to TXOd±) (Note 1)	46	54	ns
$t_{off}$	Transmit Hold Time at End of Packet (TXO±) (Note 1)	250		ns
$t_{offd}$	Transmit Hold Time at End of Packet (TXOd±) (Note 1)	200		ns

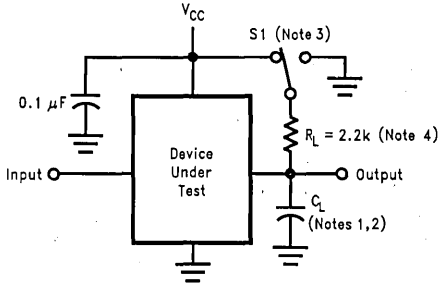
**Note 1:** This parameter is guaranteed by design and is not tested.

## 9.0 AC Timing Test Conditions

Input Pulse Levels (TTL/CMOS)	GND to 3.0V
Input Rise and Fall Times (TTL/CMOS)	5 ns
Input and Output Reference Levels (TTL/CMOS)	1.3V
Input Pulse Levels (Diff.)	-350 mV to -1315 mV
Input and Output Reference Levels (Diff.)	50% Point of the Differential
TRI-STATE Reference Levels	Float ( $\Delta V$ ) $\pm 0.5V$
Output Load (See Figure Below)	

All specifications are valid only if the mandatory isolation is employed and all differential signals are taken to be at the AUI side of the pulse transformer.

### Output Load (See Figure Below)



TL/F/11498-61

**Note 1:** Load Capacitance used depends on output type (includes scope and jig capacitance):  
 For 3SL, MOS, TPI, AUI:  $C_L = 50$  pF.  
 For 3SH, OCH:  $C_L = 240$  pF.

**Note 2:** Specifications which measure delays from an active state to a high impedance state are not guaranteed by production test, but are characterized using 70 pF, and are correlated to determine true driver turn-off time by eliminating inherent R-C delay times in measurements.

**Note 3:** S1 = Open for timing test for push pull outputs.  
 S1 =  $V_{CC}$  for  $V_{OL}$  test.  
 S1 = GND for  $V_{OH}$  test.  
 S1 =  $V_{CC}$  for High Impedance to active low and active low to High Impedance measurements.  
 = GND for High Impedance to active high and active high to High Impedance measurements.

**Note 4:** Pull-up load for CHRDY = 1 k $\Omega$ .  
 IO16 = 300 $\Omega$ .  
 M16 = 300 $\Omega$ .

## Pin Capacitance $T_A = 25^\circ C, f = 1$ MHz

Symbol	Parameter	Typ	Units
$C_{IN}$	Input Capacitance	7	pF
$C_{OUT}$	Output Capacitance	10	pF

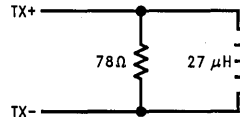
### DERATING FACTOR

Output timing is measured with a purely capacitive load of 50 pF, or 240 pF. The following correction factor can be used for other loads (**Note:** This factor is preliminary):

Derating for 3SL, MOS =  $\sim 0.05$  ns/pF

Derating for 3SH, OCL, TPI =  $\sim 0.03$  ns/pF

### AUI Transmit Test Load



TL/F/11498-62

**Note:** In the above diagram, the TX+ and TX- signals are taken from the AUI side of the isolation (pulse transformer). The pulse transformer used for all testing is a 100  $\mu H \pm 0.1\%$  Pulse Engineering PE64103.

## DP83902A ST-NIC™

### Serial Network Interface Controller for Twisted Pair

#### General Description

The DP83902A Serial Network Interface Controller for Twisted Pair (ST-NIC) is a microCMOS VLSI device designed for easy implementation of CSMA/CD local area networks. These include Ethernet (10BASE5), Thin Ethernet (10BASE2) and Twisted-pair Ethernet (10BASE-T). The overall ST-NIC solution provides the Media Access Control (MAC) and Encode-Decode (ENDEC) with an AUI interface, and 10BASE-T transceiver functions in accordance with the IEEE 802.3 standards.

The DP83902A's 10BASE-T transceiver fully complies with the IEEE standard. This functional block incorporates the receiver, transmitter, collision, heartbeat, loopback, jabber, and link integrity blocks as defined in the standard. The transceiver when combined with equalization resistors, transmit/receive filters, and pulse transformers provides a complete physical interface from the DP83902A's ENDEC module and the twisted pair medium.

The integrated ENDEC module allows Manchester encoding and decoding via a differential transceiver and phase lock loop decoder at 10 Mbit/sec. Also included are collision detect translator and diagnostic loopback capability. The ENDEC module interfaces directly to the transceiver module, and also provides a fully IEEE compliant AUI (Attachment Unit Interface) for connection to other media transceivers.

(Continued)

#### Features

- Single chip solution for IEEE 802.3, 10BASE-T
- Integrated controller, ENDEC, and transceiver
- Full AUI interface
- No external precision components required
- 3 levels of loopback supported

#### Transceiver Module

- Integrates transceiver electronics, including:
  - Transmitter and receiver
  - Collision detect, heartbeat and jabber timer
  - Link integrity test
- Link disable and polarity detection/correction
- Integrated smart receive squelch
- Reduced squelch level for extended distance cable operation (100-pin QFP version)

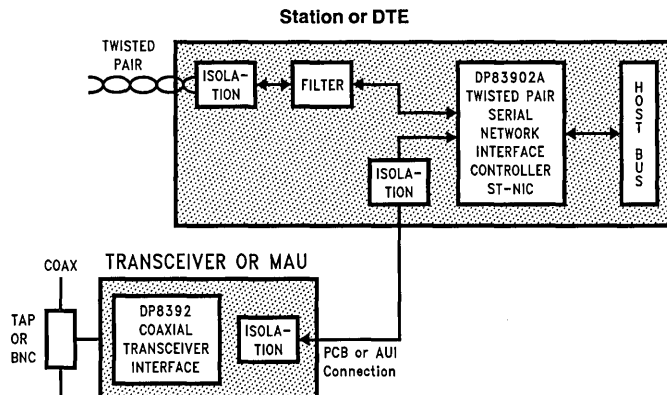
#### ENDEC Module

- 10 Mb/s Manchester encoding/decoding, plus clock recovery
- Transmitter half or full step mode
- Squelch on receive and collision pairs
- Lock time 5 bits typical
- Decodes Manchester data with up to  $\pm 18$  ns jitter

#### MAC/Controller Module

- 100% DP8390 software/hardware compatible
- Dual 16-bit DMA channels
- 16-byte internal FIFO
- Efficient buffer management implementation
- Independent system and network clocks
- Supports physical, multicast and broadcast address filtering
- Network statistics storage

#### 1.0 System Diagram



TL/F/11157-1

## General Description (Continued)

The Media Access Control function which is provided by the Network Interface Control module (NIC) provides simple and efficient packet transmission and reception control by means of unique dual DMA channels and an internal FIFO. Bus arbitration and memory control logic are integrated to reduce board cost and area overheads.

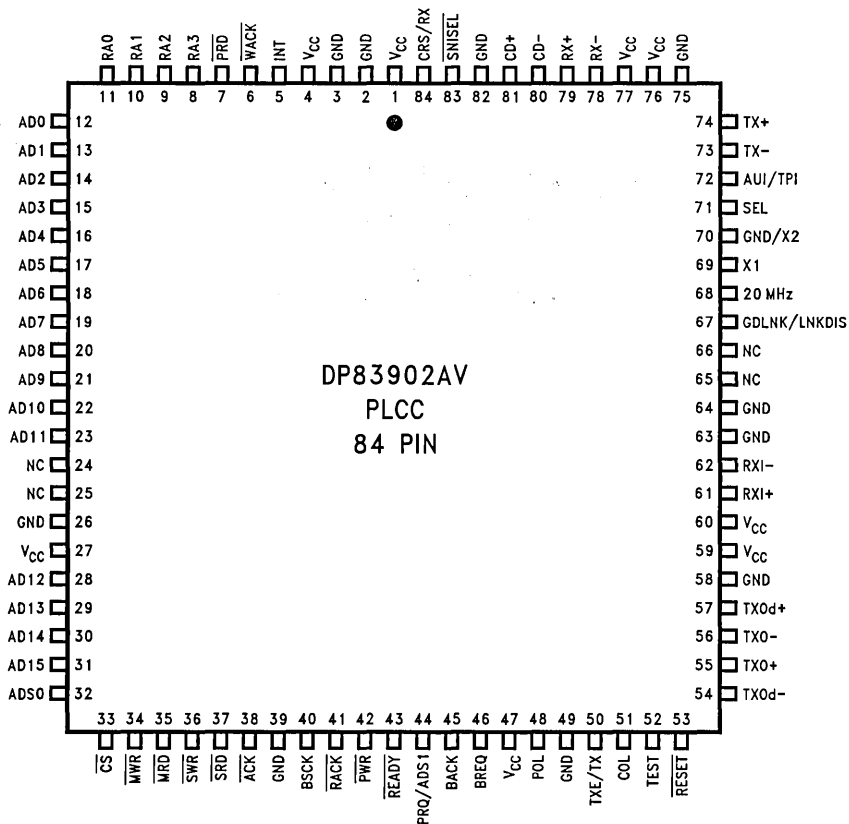
DP83902A provides a comprehensive single chip solution for 10BASE-T IEEE 802.3 networks and is designed for easy interface to other transceivers via the AUI interface.

Due to the inherent constraints of CMOS processing, isolation is required at the AUI differential signal interface for 10BASE5 and 10BASE2 applications. Capacitive or inductive isolation may be used.

## Table Of Contents

- 1.0 SYSTEM DIAGRAM
- 2.0 PIN DESCRIPTION
- 3.0 BLOCK DIAGRAM
- 4.0 FUNCTIONAL DESCRIPTION
- 5.0 TRANSMIT/RECEIVE PACKET ENCAPSULATION/DECAPSULATION
- 6.0 DIRECT MEMORY ACCESS CONTROL (DMA)
- 7.0 PACKET RECEPTION
- 8.0 PACKET TRANSMISSION
- 9.0 REMOTE DMA
- 10.0 INTERNAL REGISTERS
- 11.0 INITIALIZATION PROCEDURES
- 12.0 LOOPBACK DIAGNOSTICS
- 13.0 BUS ARBITRATION AND TIMING
- 14.0 PRELIMINARY ELECTRICAL CHARACTERISTICS
- 15.0 SWITCHING CHARACTERISTICS
- 16.0 AC TIMING TEST CONDITIONS
- 17.0 PHYSICAL DIMENSIONS

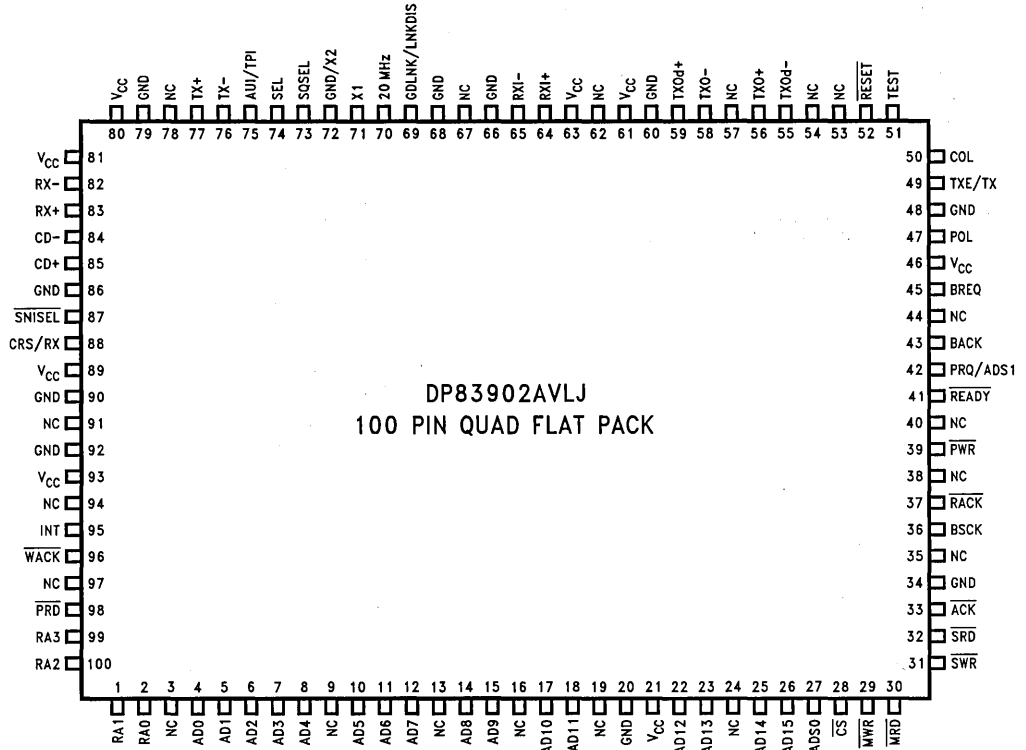
## Connection Diagrams



Order Number DP83902AV  
See NS Package Number V84A

TL/F/11157-2

Connection Diagrams (Continued)

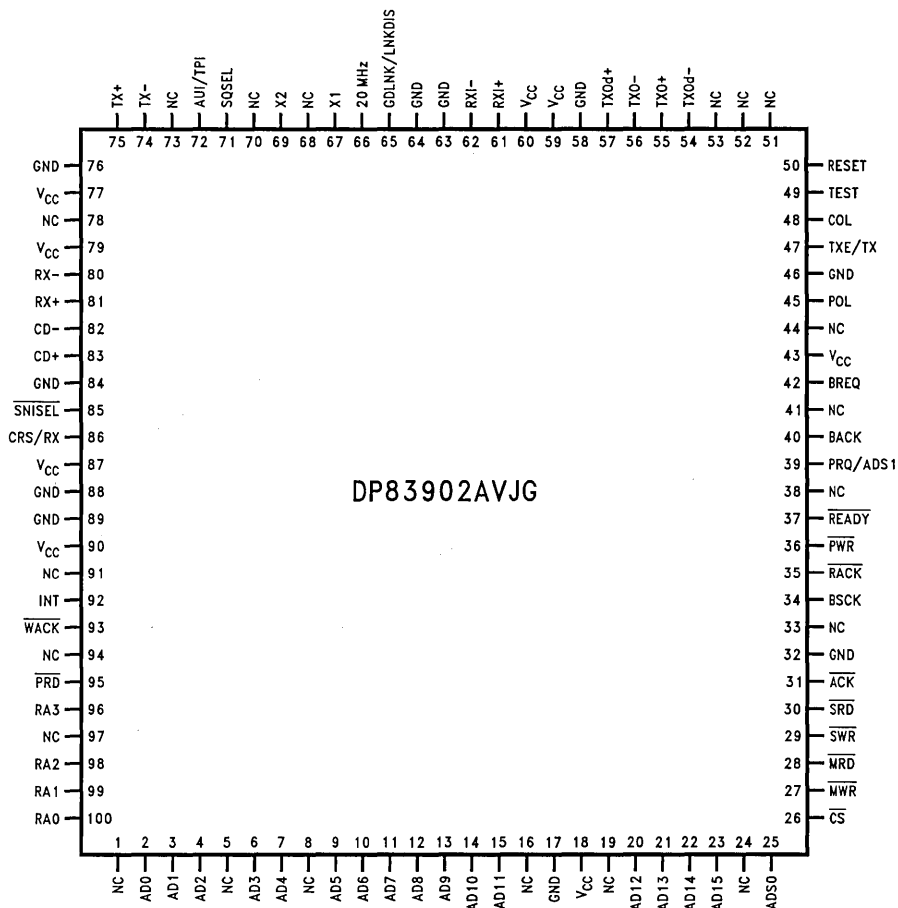


DP83902AVLJ  
100 PIN QUAD FLAT PACK

Order Number DP83902AVLJ  
See NS Package Number VLJ100A

TL/F/11157-56

## Connection Diagrams (Continued)



DP83902AVJG

Order Number DP83902AVJG  
See NS Package Number VJG100A

TL/F/11157-65

## 2.0 Pin Description

PQFP Pin No.	PLCC Pin No.	AVJG Pin No.	Pin Name	I/O	Description
<b>BUS INTERFACE PINS</b>					
95	5	92	INT	O	<b>INTERRUPT:</b> Indicates that the DP83902A requires CPU attention after reception transmission or completion of DMA transfers. The interrupt is cleared by writing to the ISR (Interrupt Status Register). All interrupts are maskable.
96	6	93	WACK	I	<b>WRITE ACKNOWLEDGE:</b> Issued from system to DP83902A to indicate that data has been written to the external latch. The DP83902A will begin a write cycle to place the data in local memory.
98	7	95	PRD	O	<b>PORT READ:</b> Enables data from external latch on to local bus during a memory write cycle to local memory (remote write operation). This allows asynchronous transfer of data from the system memory to local memory.
99, 100, 1, 2	8-11	96, 98-100	RA3-RA0	I	<b>REGISTER ADDRESS:</b> These four pins are used to select a register to be read or written. The state of these inputs is ignored when the DP83902A is not in slave mode (CS high).

1

## 2.0 Pin Description (Continued)

PQFP Pin No.	PLCC Pin No.	AVJG Pin No.	Pin Name	I/O	Description
<b>BUS INTERFACE PINS (Continued)</b>					
4-8, 10-12, 14, 15, 17, 18, 22, 23, 25, 26	12-23, 28-31	2-4, 6, 7, 9-15, 20-23	AD0- AD15	I/O, Z	<b>MULTIPLEXED ADDRESS/DATA BUS:</b> <ul style="list-style-type: none"> <li>Register Access, with DMA inactive, <math>\overline{CS}</math> low and <math>\overline{ACK}</math> returned from DP83902A, pins AD0-AD7 are used to read and write register data. AD8-AD15 float during I/O transfers, <math>\overline{SRD}</math>, <math>\overline{SWR}</math> pins are used to select direction of transfer.</li> <li>Bus Master with BACK input asserted. During t1 of memory cycle AD0-AD15 contain address. During t2, t3, t4 AD0-AD15 contain data (word transfer mode). During t2, t3, t4 AD0-AD7 contain data, AD8-AD15 contain address (byte transfer mode). Direction of transfer is indicated by DP83902A on <math>\overline{MWR}</math>, <math>\overline{MRD}</math> lines.</li> </ul>
27	32	25	ADS0	I/O, Z	<b>ADDRESS STROBE 0:</b> <ul style="list-style-type: none"> <li>Input: with DMA inactive and <math>\overline{CS}</math> low, latches RA0-RA3 inputs on falling edge. If high, data present on RA0-RA3 will flow through latch.</li> <li>Output: When Bus Master, latches address bits (AD0-AD15) to external memory during DMA transfers.</li> </ul>
28	33	26	$\overline{CS}$	I	<b>CHIP SELECT:</b> Chip Select places controller in slave mode for $\mu P$ access to internal registers. Must be valid through data portion of bus cycle. RA0-RA3 are used to select the internal register. $\overline{SWR}$ and $\overline{SRD}$ select direction of data transfer.
29	34	27	$\overline{MWR}$	O, Z	<b>MASTER WRITE STROBE:</b> (Strobe for DMA transfers) Active low during write cycles (t2, t3, tw) to buffer memory. Rising edge coincides with the presence of valid output data. TRI-STATE® until BACK asserted.
30	35	28	$\overline{MRD}$	O, Z	<b>MASTER READ STROBE:</b> (Strobe for DMA transfers) Active during read cycles (t2, t3, tw) to buffer memory. Input data must be valid on rising edge of $\overline{MRD}$ . TRI-STATE until BACK asserted.
31	36	29	$\overline{SWR}$	I	<b>SLAVE WRITE STROBE:</b> Strobe from CPU to write an internal register selected by RA0-RA3. Data is latched into the DP83902A on the rising edge of this input.
32	37	30	$\overline{SRD}$	I	<b>SLAVE READ STROBE:</b> Strobe from CPU to read an internal register selected by RA0-RA3. The register data is output when $\overline{SRD}$ goes low.
33	38	31	$\overline{ACK}$	O	<b>ACKNOWLEDGE:</b> Active low when DP83902A grants access to CPU. Used to insert WAIT states to CPU until DP83902A is synchronized for a register read or write operation.
36	40	34	BSCK	I	<b>BUS CLOCK:</b> This clock is used to establish the period of the DMA memory cycle. Four clock cycles (t1, t2, t3, t4) are used per DMA cycle. DMA transfers can be extended by one BSCK increment using the $\overline{READY}$ input.
37	41	35	$\overline{RACK}$	I	<b>READ ACKNOWLEDGE:</b> Indicates that the system DMA or host CPU has read the data placed in the external latch by the DP83902A. The DP83902A will begin a read cycle to update the latch.
39	42	36	$\overline{PWR}$	O	<b>PORT WRITE:</b> Strobe used to latch data from the DP83902A into external latch for transfer to host memory during Remote Read transfers. The rising edge of $\overline{PWR}$ coincides with the presence of valid data on the local bus.
41	43	37	$\overline{READY}$	I	<b>READY:</b> This pin is set high to insert wait states during a DMA transfer. The DP83902A will sample this signal at t3 during DMA transfers.
42	44	39	PRQ/ ADS1	O, Z	<b>PORT REQUEST/ADDRESS STROBE 1</b> <ul style="list-style-type: none"> <li>32-BIT MODE: If LAS is set in the Data Configuration Register, this line is programmed as ADS1. It is used to strobe addresses A16-A31 into external latches. (A16-A31 are the fixed addresses stored in RSAR0, RSAR1). ADS1 will remain at TRI-STATE until BACK is received.</li> <li>16-BIT MODE: If LAS is not set in the Data Configuration Register, this line is programmed as PRQ and is used for Remote DMA Transfers. The DP83902A initiates a single remote DMA read or write operation by asserting this pin. In this mode PRQ will be a standard logic output.</li> </ul> <p>Note: This line will power up as TRI-STATE until the Data Configuration Register is programmed.</p>



## 2.0 Pin Description (Continued)

PQFP Pin No.	PLCC Pin No.	AVJG Pin No.	Pin Name	I/O	Description
<b>BUS INTERFACE PINS (Continued)</b>					
43	45	40	BACK	I	<b>BUS ACKNOWLEDGE:</b> Bus Acknowledge is an active high signal indicating that the CPU has granted the bus to the DP83902A. If immediate bus access is desired, BREQ should be tied to BACK. <b>Tying BACK to V<sub>CC</sub> will result in a deadlock.</b>
45	46	42	BREQ	O	<b>BUS REQUEST:</b> Bus Request is an active high signal used to request the bus for DMA transfers. This signal is automatically generated when the FIFO needs servicing.
52	53	50	RESET	I	<b>RESET:</b> Reset is active low and places the DP83902A in a reset mode immediately. No packets are transmitted or received by the DP83902A until STA bit is set. Affects Command Register, Interrupt Mask Register, Data Configuration Register and Transmit Configuration Register. The DP83902A will execute reset within 10 BSCK cycles.
<b>NETWORK INTERFACE PINS</b>					
47	48	45	POL	O	<b>POLARITY:</b> A TTL/MOS active high output. This signal is normally in the low state. When the TPI module detects seven consecutive link pulses or three consecutive received packets with reversed polarity POL, is asserted.
49	50	47	TXE/TX	O	<b>TRANSMIT ENABLE/TRANSMIT:</b> A TTL/MOS active high output. It is asserted for approximately 50 ms whenever the DP83902A transmits data in either AUI or TPI modes.
50	51	48	COL	O	<b>COLLISION:</b> A TTL/MOS active high output. It is asserted for approximately 50 ms whenever the DP83902A detects a collision in either the AUI or TPI modes.
51	52	49	TEST	I	<b>FACTORY TEST INPUT:</b> Used to check the chip's internal functions. This should be tied low during normal operation.
55, 56, 58, 59	54, 55, 56, 57	54, 55, 56, 57	TXO <sub>d-</sub> , TXO <sub>+</sub> , TXO <sub>-</sub> , TXO <sub>d+</sub>	O	<b>TWISTED PAIR TRANSMIT OUTPUTS:</b> These high drive CMOS level outputs are resistively combined external to the chip to produce a differential output signal with equalization to compensate for Intersymbol Interference (ISI) on the twisted pair medium.
64, 65	61, 62	61, 62	RXI <sub>+</sub> , RXI <sub>-</sub>	I	<b>TWISTED PAIR RECEIVE INPUTS:</b> These inputs feed a differential amplifier which passes valid data to the ENDEC module.
69	67	65	GDLNK/ LNKDIS	I/O	<b>GOOD LINK/LINK DISABLE:</b> This pin has a dual function both input and output. The function is latched by the DP83902A on the rising edge of the Reset signal i.e.: on the chip returning to normal operation after reset. As an output this pin is configured as an open drain N-channel device and is suitable for driving a LED. It will be latched as output on removal of chip reset if connected to a LED or left open circuit. Under normal conditions (the twisted pair link is not broken) the output will be low, and the LED will be lit. The open drain output will be switched off if the twisted pair link has been detected to be broken. It is recommended that the color of the LED be green. This output will be pulled high in AUI mode, by an internal resistor of approximately 15 k $\Omega$ . When this pin, which has an internal pull-up resistor to V <sub>DD</sub> , is tied low it becomes an input and the link integrity checking is disabled.
73	—	70	SQSEL	I	<b>TPI SQUELCH SELECT:</b> This pin selects the TPI module input squelch thresholds. When tied low, the input squelch threshold on the RXI $\pm$ inputs complies to 10BASE-T specification. When set high, the RXI $\pm$ input operates with reduced squelch levels, allowing its use with longer lengths of cable or cable with higher losses. If this pin is left unconnected, an internal pulldown causes the ST-NIC's TPI to default to the higher squelch level.
70	68	66	20 MHz	O	<b>20 MHz:</b> This is a TTL/MOS level signal. It is a buffered version of the oscillator X2. It is suitable to drive external logic.
71	69	67	X1	I	<b>EXTERNAL OSCILLATOR INPUT</b>
72	70	69	GND/ X2	I	<b>GROUND/X2:</b> If an oscillator is used, this pin should be tied to ground and if a crystal is used, this pin should be tied directly to the crystal.
74	71	71	SEL	I	<b>MODE SELECT:</b> When high, TX <sub>+</sub> and TX <sub>-</sub> are the same voltage in the idle state. When low, Transmit <sub>+</sub> is positive with respect to Transmit <sub>-</sub> in the idle state, at the transformer's primary.

## 2.0 Pin Description (Continued)

PQFP Pin No.	PLCC Pin No.	AVJG Pin No.	Pin Name	I/O	Description
<b>NETWORK INTERFACE PINS (Continued)</b>					
75	72	72	AUI/ TPI	I	<b>AUI/TPI SELECT:</b> A TTL level active high input that selects either the AUI interface or the TPI module for interface with the ENDEC module. When high the AUI is selected, when low the TPI is selected.
76, 77	73, 74	74, 75	TX-, TX+	O	<b>AUI TRANSMIT OUTPUT:</b> Differential driver which sends the encoded data to the transceiver. The outputs are source followers which require 270Ω pull-down resistors.
82, 83	78, 79	80, 81	RX-, RX+	I	<b>AUI RECEIVE INPUT:</b> Differential receive input pair from the transceiver.
84, 85	80, 81	82, 83	CD-, CD+	I	<b>AUI COLLISION INPUT:</b> Differential collision pair input from the transceiver.
87	83	85	SNISEL	I	<b>FACTORY TEST INPUT:</b> For normal operation tied to V <sub>CC</sub> . When low enables the ENDEC module to be tested independently of the DP83902A module.
88	84	86	CRS/ RX	O	<b>CARRIER SENSE/RECEIVE:</b> A TTL/MOS level active high signal. It is asserted for approximately 50 ms whenever valid transmit or receive data is detected while in AUI mode or receive data is detected while in TPI mode.
<b>POWER SUPPLY PINS (DIGITAL)</b>					
21, 46, 89	1, 27, 47	18, 43, 87	V <sub>CC</sub>		<b>POSITIVE 5V SUPPLY PINS</b>
20, 34, 48, 68, 90	2, 26, 39, 49, 64	17, 32, 46, 64, 88	GND		<b>NEGATIVE (GROUND) SUPPLY PINS:</b> It is suggested that a decoupling capacitor be connected between the V <sub>CC</sub> and GND pins.
<b>POWER SUPPLY PINS (ANALOG)</b>					
93	4	90	V <sub>CC</sub>		<b>VCO 5V SUPPLY PIN:</b> Care should be taken to reduce noise on this pin as it supplies power to the analog VCO to the Phase Lock Loop.
92	3	89	GND		<b>VCO GROUND SUPPLY PIN:</b> Care should be taken to reduce noise on this pin as it supplies ground to the analog VCO to the Phase Lock Loop.
63	60	60	V <sub>CC</sub>		<b>TPI RECEIVE 5V SUPPLY:</b> Power pin supplies 5V to the Twisted Pair Interface Receiver.
66	63	63	GND		<b>TPI RECEIVE GROUND:</b> Ground pin for the Twisted Pair Interface Receiver.
61	59	59	V <sub>CC</sub>		<b>TPI TRANSMIT 5V SUPPLY:</b> Power pin supplies 5V to the Twisted Pair Interface Transmitter.
60	58	58	GND		<b>TPI TRANSMIT GROUND:</b> Ground pin for the Twisted Pair Interface Transmitter.
81	77	79	V <sub>CC</sub>		<b>AUI RECEIVE 5V SUPPLY:</b> Power pin supplies 5V to the AUI Interface Receiver.
86	82	84	GND		<b>AUI RECEIVE GROUND:</b> Ground pin for the AUI Interface Receiver.
80	76	77	V <sub>CC</sub>		<b>AUI TRANSMIT 5V SUPPLY:</b> Power pin supplies 5V to AUI Interface Transmitter.
79	75	76	GND		<b>AUI TRANSMIT GROUND:</b> Ground pin for the AUI Interface Transmitter.
<b>NO CONNECTION</b>					
3, 9, 13, 16, 19, 24, 35, 38, 40, 44, 53, 54, 57, 62, 67, 78, 91, 94, 97	24, 25, 65, 66	1, 5, 8, 16, 19, 24, 33, 38, 41, 44, 51, 52, 53, 68, 70, 73, 78, 91, 94, 97	NC		NO CONNECTION. Do not connect to these pins.

### 3.0 Block Diagram

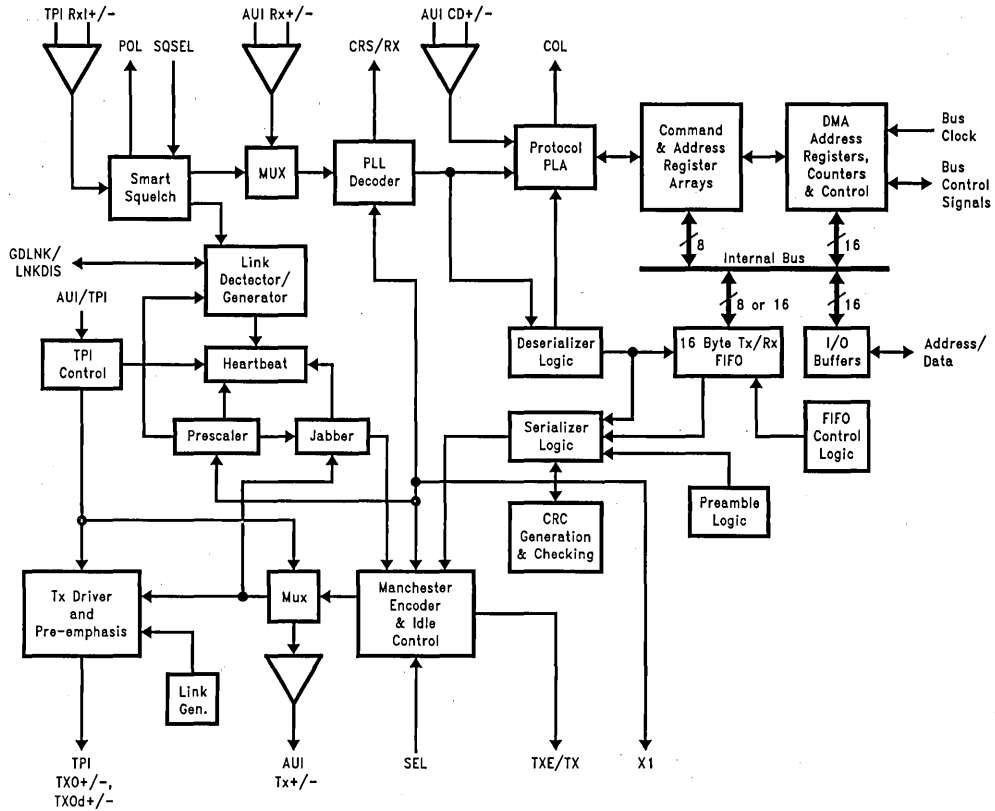
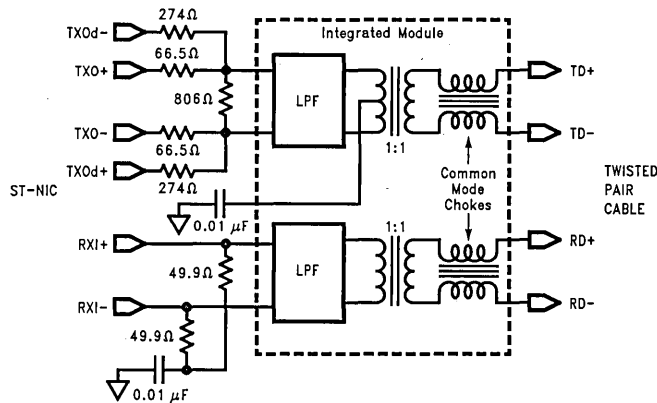


FIGURE 1

TL/F/11157-3

### Typical Connection to Twisted Pair Cable



TL/F/11157-4

- Recommended integrated modules are:
- 1) Pulse Engineering PE65431
  - 2) Beltuse 0556-2006-01 or 0556-3392-00
  - 3) Valor FL1012.

### ST-NIC Twisted Pair Interface

## 4.0 Functional Description (Refer to Figure 1)

### TWISTED PAIR INTERFACE (TPI) MODULE

The TPI consists of five main logical functions:

- The Smart Squelch, responsible for determining when valid data is present on the differential receive inputs (RXI $\pm$ ).
- The Collision function checks for simultaneous transmission and reception of data on the TXO $\pm$  and RXI $\pm$  pins.
- The Link Detector/Generator checks the integrity of the cable connecting the two twisted pair MAUs.
- The Jabber disables the transmitter if it attempts to transmit a longer than legal packet.
- The Tx Driver & Pre-emphasis transmits Manchester encoded data to the twisted pair network via the summing resistors and transformer/filter.

#### SMART SQUELCH

The ST-NIC implements an intelligent receive squelch on the RXI $\pm$  differential inputs to ensure that impulse noise on the receive inputs will not be mistaken for a valid signal.

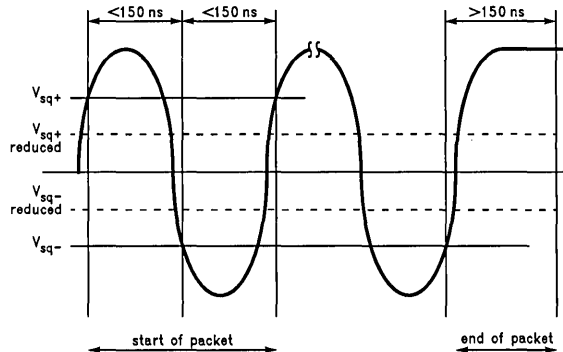
The squelch circuitry employs a combination of amplitude and timing measurements to determine the validity of data on the twisted pair inputs. There are two squelch levels which are selectable via the SQSEL pin. One mode is 10BASE-T compatible, and the second is reduced squelch mode.

The diagram shows the 10BASE-T mode operation of the smart squelch.

The signal at the start of packet is checked by the smart squelch and any pulses not exceeding the squelch level (either positive or negative, depending upon polarity) will be rejected. Once this first squelch level is overcome correctly the opposite squelch level must then be exceeded within 150 ns. Finally the signal must exceed the original squelch level within a further 150 ns to ensure that the input waveform will not be rejected. The checking procedure results in the loss of typically three bits at the beginning of each packet.

Only after all these conditions have been satisfied will a control signal be generated to indicate to the remainder of the circuitry that valid data is present. At this time the smart squelch circuitry is reset.

Valid data is considered to be present until either squelch level has not been generated for a time longer than 150 ns, indicating End of Packet. Once good data has been detected the squelch levels are reduced to minimize the effect of noise causing premature End of Packet detection.



TL/F/11157-5

The reduced squelch mode functions the same as the 10BASE-T mode except that only the lower level is used for both turn-on and turn-off.

#### COLLISION

A collision is detected by the TPI module when the receive and transmit channels are active simultaneously. If the TPI is receiving when a collision is detected it is reported to the controller immediately. If, however, the TPI is transmitting when a collision is detected the collision is not reported until seven bits have been received while in the collision state. This prevents a collision being reported incorrectly due to noise on the network. The signal to the controller remains for the duration of the collision.

Approximately  $1 \mu\text{s}$  after the transmission of each packet a signal called the Signal Quality Error (SQE) consisting of typically 10 cycles of 10 MHz is generated. This 10 MHz signal, also called the Heartbeat, ensures the continued functioning of the collision circuitry.

#### LINK DETECTOR/GENERATOR

The link generator is a timer circuit that generates a link pulse as defined by the 10BASE-T specification that will be generated by the transmitter section. The pulse which is 100 ns wide is transmitted on the TXO+ output, every 16 ms, in the absence of transmit data.

The pulse is used to check the integrity of the connection to the remote MAU. The link detection circuit checks for valid pulses from the remote MAU and if valid link pulses are not received the link detector will disable the transmit, receive and collision detection functions.

The GDLNK output can directly drive a LED to show that there is a good twisted pair link. For normal conditions the LED will be on. The link integrity function can be disabled as described in the Pin Description Section.

#### JABBER

The jabber timer monitors the transmitter and disables the transmission if the transmitter is active for greater than 26 ms. The transmitter is then disabled for the whole time that the ENDEC module's internal transmit enable is asserted. This signal has to be deasserted for approximately 750 ms (the unjab time) before the Jabber re-enables the transmit outputs.

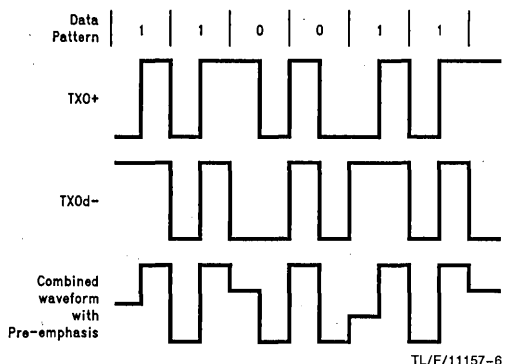
#### TRANSMIT DRIVER

The transmitter consists of four signals, the true and complement Manchester encoded data (TXO $\pm$ ) and these signals delayed by 50 ns (TXOd $\pm$ ).

## 4.0 Functional Description (Continued)

These four signals are resistively combined, TXO+ with TXOd- and TXO- with TXOd+. This is known as digital pre-emphasis and is required to compensate for the twisted pair cable which acts like a low pass filter causing greater attenuation to the 10 MHz (50 ns) pulses of the Manchester encoded waveform than the 5 MHz (100 ns) pulses.

An example of how these signals are combined is shown in the following diagram.



The signal with pre-emphasis shown above is generated by resistively combining TXO+ and TXOd-. This signal along with its complement is passed to the transmit filter.

### STATUS INFORMATION

Status information is provided by the ST-NIC on the CRS/RX, TXE/TX, COL and POL outputs as described in the pin description table. These outputs are suitable for driving status LEDs via an appropriate driver circuit.

The POL output is normally low, and will be driven high when seven consecutive link pulses or three consecutive receive packets are detected with reversed polarity. A polarity reversal can be caused by a wiring error at either end of the TPI cable. On detection of a polarity reversal the condition is latched and POL is asserted. The TPI corrects for this error internally and will decode received data correctly, eliminating the need to correct the wiring error.

### ENCODER/DECODER (ENDEC) MODULE

The ENDEC consists of three main logical blocks:

- The Manchester encoder accepts NRZ data from the controller, encodes the data to Manchester, and transmits it differentially to the transceiver, through the differential transmit driver.
- The Manchester decoder receives Manchester data from the transceiver, converts it to NRZ data and clock pulses, and sends it to the controller.
- The collision translator indicates to the controller the presence of a valid 10 MHz collision signal to the PLL.

### MANCHESTER ENCODER AND DIFFERENTIAL DRIVER

The differential transmit pair, on the secondary of the transformer, drives up to 50 meters of twisted pair AUI cable. These outputs are source followers which require two 270 $\Omega$  pull-down resistors to ground.

The DP83902A allows both half-step and full-step to be compatible with Ethernet and IEEE 802.3. With the SEL pin low (for Ethernet I). Transmit+ is positive with respect to

Transmit- during idle; with SEL high (for IEEE 802.3), Transmit+ and Transmit- are equal in the idle state. This provides zero differential voltage to operate with transformer coupled loads.

### MANCHESTER DECODER

The decoder consists of a differential receiver and a PLL to separate a Manchester decoded data stream into internal clock signals and data. The differential input must be externally terminated with two 39 $\Omega$  resistors connected in series if the standard 78 $\Omega$  transceiver drop cable is used. In thin Ethernet applications, these resistors are optional. To prevent noise from falsely triggering the decoder, a squelch circuit at the input rejects signals with levels less than -175 mV. Signals more negative than -300 mV are decoded. Data becomes valid typically within 5 bit times. The DP83902A may tolerate bit jitter up to 18 ns in the received data. The decoder detects the end of a frame when no more mid-bit transitions are detected.

### COLLISION TRANSLATOR

When in AUI mode, when the Ethernet transceiver (DP8392 CT) detects a collision, it generates a 10 MHz signal to the differential collision inputs (CD $\pm$ ) of the DP83902A. When these inputs are detected active, the DP83902A uses this signal to back off its current transmission and reschedule another one.

The collision differential inputs are terminated the same way as the differential receive inputs. The squelch circuitry is also similar, rejecting pulses with levels less than -175 mV.

### CRYSTAL/OSCILLATOR OPERATION

#### OSCILLATOR

The oscillator is controlled by a 20 MHz parallel resonant crystal connected between X1 and X2 or by an external clock on X1. The 20 MHz output of the oscillator is divided by 2 to generate the 10 MHz transmit clock for the controller. The oscillator also provides internal clock signals to the encoding and decoding circuits.

**Note:** When X1 is being driven by an external oscillator, X2 MUST be grounded.

#### Crystal Specifications

Resonant Frequency	20 MHz
Tolerance	$\pm 0.005\%$ at 25°C
Stability	$\pm 0.005\%$ at 0°C-70°C
Type	AT Cut
Circuit	Parallel Resonance
Max ESR	25 $\Omega$
Crystal Load Capacitor	20 pF

The 20 MHz crystal connection to the DP83902 requires special care. The IEEE 802.3 standard requires the transmitted signal frequency to be accurate within  $\pm 0.01\%$ . Stray capacitance can shift the crystal's frequency out of range and cause transmitted frequency to exceed its 0.01% tolerance. The frequency marked on the crystal is usually measured with a fixed load capacitance specified in the crystal's data sheet, typically 20 pF.

## 4.0 Functional Description (Continued)

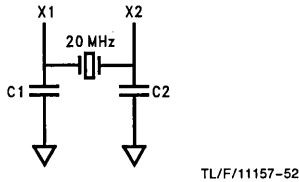
In order to prevent distortion on the transmitted frequency, the total capacitance seen by the crystal should equal the total load capacitance. On a standard parallel set-up as shown in the diagram below, the 2 load caps C1 and C2 should equal 2C1, the spec load cap, (due to the capacitors acting in series) less any stray capacitances.

Thus the trim capacitors required can be calculated as follows:

$C1 = 2XC1 - (Cb1 + Cd1)$  Where Cb1 = Board cap on X1 and Cd1 = X1 dev cap

$C2 = 2XC1 - (Cb2 + Cd2)$  Where Cb2 = Board cap on X2 and Cd2 = X2 dev cap

The value of STNIC pins X1 and X2 is in the region of 5 pF.



### NIC (Media Access Control) MODULE

#### RECEIVE DESERIALIZER

The Receive Deserializer is activated when the input signal Carrier Sense is asserted to allow incoming bits to be shifted into the shift register by the receive clock. The serial receive data is also routed to the CRC generator/checker. The Receive Deserializer includes a synch detector which detects the SFD (Start of Frame Delimiter) to establish where byte boundaries within the serial bit stream are located. After every eight receive clocks, the byte wide data is transferred to the 16-byte FIFO and the Receive Byte Count is incremented. The first six bytes after the SFD are checked for valid comparison by the Address Recognition Logic. If the Address Recognition Logic does not recognize the packet, the FIFO is cleared.

#### CRC GENERATOR/CHECKER

During transmission, the CRC logic generates a local CRC field for the transmitted bit sequence. The CRC encodes all fields after the SFD. The CRC is shifted out MSB first following the last transmit byte. During reception the CRC logic generates a CRC field from the incoming packet. This local CRC is serially compared to the incoming CRC appended to the end of the packet by the transmitting node. If the local and received CRC match, a specific pattern will be generated and decoded to indicate no data errors. Transmission errors result in different pattern and are detected, resulting in rejection of a packet (if so programmed).

#### TRANSMIT SERIALIZER

The Transmit Serializer reads parallel data from the FIFO and serializes it for transmission. The serializer is clocked by the transmit clock generated internally. The serial data is also shifted into the CRC generator/checker. At the beginning of each transmission, the Preamble and Synch Generator append 62 bits of 1,0 preamble and a 1,1 synch pattern. After the last data byte of the packet has been serialized the 32-bit FCS field is shifted directly out of the CRC generator. In the event of a collision the Preamble and Synch generators are used to generate a 32-bit JAM pattern of all 1's.

#### ADDRESS RECOGNITION LOGIC

The address recognition logic compares the Destination Address Field (first 6 bytes of the received packet) to the Physical address registers stored in the Address Register Array. If any one of the six bytes does not match the pre-programmed physical address, the Protocol Control Logic rejects the packet. All multicast destination addresses are filtered using a hashing technique. (See register description.) If the multicast address indexes a bit that has been set in the filter bit array of the Multicast Address Register Array the packet is accepted, otherwise it is rejected by the Protocol Control Logic. Each destination address is also checked for all 1's which is the reserved broadcast address.

#### FIFO AND BUS OPERATIONS

##### Overview

To accommodate the different rates at which data comes from (or goes to) the network and goes to (or comes from) the system memory, the ST-NIC contains a 16-byte FIFO for buffering data between the media. The FIFO threshold is programmable. When the FIFO has filled to its programmed threshold, the local DMA channel transfers these bytes (or words) into local memory. It is crucial that the local DMA is given access to the bus within a minimum bus latency time; otherwise a FIFO underrun (or overrun) occurs.

FIFO underruns or overruns are caused by two conditions: (1) the bus latency is so long that the FIFO has filled (or emptied) from the network before the local DMA has serviced the FIFO and (2) the bus latency has slowed the throughput of the local DMA to a point where it is slower than the network data rate (10 Mbit/sec). This second condition is also dependent upon DMA clock and word width (byte wide or word wide). The worst case condition ultimately limits the overall bus latency which the ST-NIC can tolerate.

##### Beginning of Receive

At the beginning of reception, the ST-NIC stores the entire Address field of each incoming packet in the FIFO to determine whether the address matches the ST-NIC's Physical Address Registers or maps to one of its Multicast Registers. This causes the FIFO to accumulate 8 bytes. Furthermore, there are some synchronization delays in the DMA PLA. Thus, the actual time to when BREQ is asserted from the time the Start of Frame Delimiter (SFD) is detected is 7.8  $\mu$ s. This operation affects the bus latencies at 2- and 4-byte thresholds during the first receive BREQ since the FIFO must be filled to 8 bytes (or 4 words) before issuing a BREQ.

##### End of Receive

When the end of a packet is detected by the ENDEC module, the ST-NIC enters its end of packet processing sequence, emptying its FIFO and writing the status information at the beginning of the packet. The ST-NIC holds onto the bus for the entire sequence. The longest time BREQ may be extended occurs when a packet ends just as the ST-NIC performs its last FIFO burst. The ST-NIC, in this case, performs a programmed burst transfer followed by flushing the remaining bytes in the FIFO, and completes by writing the header information to memory. The following steps occur during this sequence.

1. ST-NIC issues BREQ because the FIFO threshold has been reached.
2. During the burst, packet ends, resulting in BREQ extended.

### 4.0 Functional Description (Continued)

3. ST-NIC flushes remaining bytes from FIFO.
4. ST-NIC performs internal processing to prepare for writing the header.
5. ST-NIC writes 4-byte (2-word) header.
6. ST-NIC de-asserts BREQ.

#### FIFO Threshold Detection

To assure that no overwriting of data in the FIFO, the FIFO logic flags a FIFO overrun as the 13th byte is written into the FIFO, effectively shortening the FIFO to 13 bytes. The FIFO logic also operates differently in Byte Mode and in Word Mode. In Byte Mode, a threshold is indicated when the  $n + 1$  byte has entered the FIFO; thus, with an 8-byte threshold, the ST-NIC issues Bus Request (BREQ) when the 9th byte has entered the FIFO. For Word Mode, BREQ is not generated until  $n + 2$  bytes have entered the FIFO. Thus, with a 4 word threshold (equivalent to an 8-byte threshold), BREQ is issued when the 10th byte has entered the FIFO.

#### Beginning of Transmit

Before transmitting, the ST-NIC performs a prefetch from memory to load the FIFO. The number of bytes prefetched is the programmed FIFO threshold. The next BREQ is not issued until after the ST-NIC actually begins transmitting data, i.e., after SFD.

#### Reading the FIFO

During normal operation, the FIFO must not be read. The ST-NIC will not issue an ACKnowledge back to the CPU if the FIFO is read. The FIFO should only be read during loop-back diagnostics.

#### PROTOCOL PLA

The protocol PLA is responsible for implementing the IEEE 802.3 protocol, including collision recovery with random backoff. The Protocol PLA also formats packets during transmission and strips preamble and synch during reception.

#### DMA AND BUFFER CONTROL LOGIC

The DMA and Buffer Control Logic is used to control two 16-bit DMA channels. During reception, the local DMA stores packets in a receive buffer ring, located in buffer memory. During transmission the Local DMA uses programmed pointer and length registers to transfer a packet from local buffer memory to the FIFO. A second DMA channel is used as a slave DMA to transfer data between the local buffer memory and the host system. The Local DMA and Remote DMA are internally arbitrated, with the Local DMA channel having highest priority. Both DMA channels use a common external bus clock to generate all required

bus timing. External arbitration is performed with a standard bus request, bus acknowledge handshake protocol.

### 5.0 Transmit/Receive Packet Encapsulation/Decapsulation

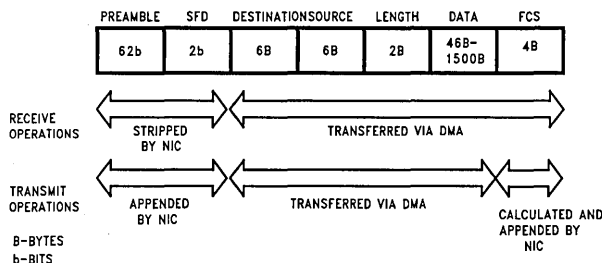
A standard IEEE 802.3 packet consists of the following fields: preamble, Start of Frame Delimiter (SFD), destination address, source address, length, data, and Frame Check Sequence (FCS). The typical format is shown in *Figure 2*. The packets are Manchester encoded and decoded by the ENDEC module and transferred serially to the NIC module using NRZ data with a clock. All fields are of fixed length except for the data field. The ST-NIC generates and appends the preamble, SFD and FCS field during transmission. The Preamble and SFD fields are stripped during reception. (The CRC is passed through to buffer memory during reception.)

#### PREAMBLE AND START OF FRAME DELIMITER (SFD)

The Manchester encoded alternating 1,0 preamble field is used by the ENDEC to acquire bit synchronization with an incoming packet. When transmitted each packet contains 62 bits of alternating 1,0 preamble. Some of this preamble will be lost as the packet travels through the network. The preamble field is stripped by the NIC module. Byte alignment is performed with the Start of Frame Delimiter (SFD) pattern which consists of two consecutive 1's. The ST-NIC does not treat the SFD pattern as a byte, it detects only the two bit pattern. This allows any preceding preamble within the SFD to be used for phase locking.

#### DESTINATION ADDRESS

The destination address indicates the destination of the packet on the network and is used to filter unwanted packets from reaching a node. There are three types of address formats supported by the ST-NIC: physical, multicast and broadcast. The physical address is a unique address that corresponds only to a single node. All physical addresses have an MSB of "0". These addresses are compared to the internally stored physical address registers. Each bit in the destination address must match in order for the ST-NIC to accept the packet. Multicast addresses begin with an MSB of "1". The ST-NIC filters multicast addresses using a standard hashing algorithm that maps all multicast addresses into a 6-bit value. This 6-bit value indexes a 64-bit array that filters the value. If the address consists of all 1's it is a broadcast address, indicating that the packet is intended for all nodes. A promiscuous mode allows reception of all packets: the destination address is not required to match any filters. Physical, broadcast, multicast, and promiscuous address modes can be selected.



TL/F/11157-7

FIGURE 2

## 5.0 Transmit/Receive Packet Encapsulation/Decapsulation

(Continued)

### SOURCE ADDRESS

The source address is the physical address of the node that sent the packet. Source addresses cannot be multicast or broadcast addresses. This field is simply passed to buffer memory.

### LENGTH/TYPE FIELD

The 2-byte length field indicates the number of bytes that are contained in the data field of the packet. This field is not interpreted by the ST-NIC.

### DATA FIELD

The data field consists of anywhere from 46 to 1500 bytes. Messages longer than 1500 bytes need to be broken into multiple packets. Messages shorter than 46 bytes will require appending a pad to bring the data field to the minimum length of 46 bytes. If the data field is padded, the number of valid data bytes is indicated in the length field. **The ST-NIC does not strip or append pad bytes for short packets, or check for oversize packets.**

### FCS FIELD

The Frame Check Sequence (FCS) is a 32-bit CRC field calculated and appended to a packet during transmission to allow detection of errors when a packet is received. During reception, error free packets result in a specific pattern in the CRC generator. Packets with improper CRC will be rejected. The AUTODIN II ( $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X^1 + 1$ ) polynomial is used for the CRC calculations.

## 6.0 Direct Memory Access Control (DMA)

The DMA capabilities of the ST-NIC greatly simplify the use of the DP83902A in typical configurations. The local DMA channel transfers data between the FIFO and memory. On transmission, the packet is DMAed from memory to the FIFO in bursts. Should a collision occur (up to 15 times), the packet is retransmitted with no processor intervention. On reception, packets are DMAed from the FIFO to the receive buffer ring (as explained below).

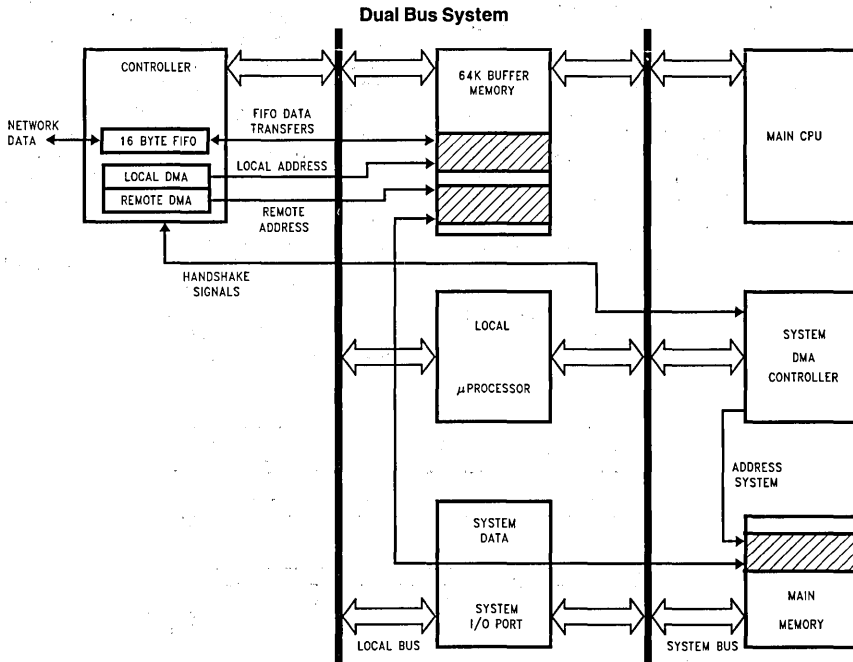
dA remote DMA channel is also provided on the ST-NIC to accomplish transfers between a buffer memory and system memory. The two DMA channels can alternatively be combined to form a single 32-bit address with 8- or 16-bit data.

### DUAL DMA CONFIGURATION

An example configuration using both the local and remote DMA channels is shown below. Network activity is isolated on a local bus, where the ST-NIC's local DMA channel performs burst transfers between the buffer memory and the ST-NIC's FIFO. The Remote DMA transfers data between the buffer memory and the host memory via a bidirectional I/O port. The Remote DMA provides local addressing capability and is used as a slave DMA by the host. Host side addressing must be provided by a host DMA or the CPU. The ST-NIC allows Local and Remote DMA operations to be interleaved.

### SINGLE CHANNEL DMA OPERATION

If desirable, the two DMA channels can be combined to provide a 32-bit DMA address. The upper 16 bits of the 32-bit address are static and are used to point to a 64 kbyte (or 32k word) page of memory where packets are to be received and transmitted.



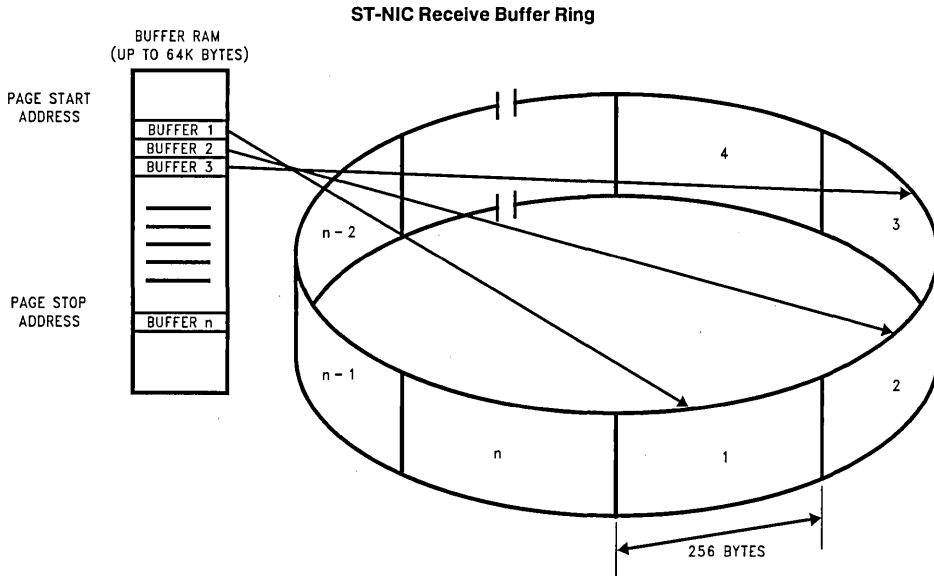


## 7.0 Packet Reception

The Local DMA receive channel uses a Buffer Ring Structure comprised of a series of contiguous fixed length 256-byte (128 word) buffers for storage of received packets. The location of the Receive Buffer Ring is programmed in two registers, a Page Start and a Page Stop Register. Ethernet packets consist of a distribution of shorter link control packets and longer data packets, the 256-byte buffer length provides a good compromise between short packets and longer packets to most efficiently use memory. In addition these buffers provide memory resources for storage of back-to-back packets in loaded networks. The assignment of buffers for storing packets is controlled by Buffer Management Logic in the ST-NIC. The Buffer Management Log-

ic provides three basic functions: linking receive buffers for long packets, recovery of buffers when a packet is rejected, and recirculation of buffer pages that have been read by the host.

At initialization, a portion of the 64 kbyte (or 32k word) address space is reserved for the receive buffer ring. Two 8-bit registers, the Page Start Address Register (PSTART) and the Page Stop Address Register (PSTOP) define the physical boundaries of where the buffers reside. The ST-NIC treats the list of buffers as a logical ring; whenever the DMA address reaches the Page Stop Address, the DMA is reset to the Page Start Address.



TL/F/11157-9

## 7.0 Packet Reception (Continued)

### INITIALIZATION OF THE BUFFER RING

Two static registers and two working registers control the operation of the Buffer Ring. These are the Page Start Register, Page Stop Register (both described previously), the Current Page Register and the Boundary Pointer Register. The Current Page Register points to the first buffer used to store a packet and is used to restore the DMA for writing status to the Buffer Ring or for restoring the DMA address in the event of a Runt packet, a CRC, or Frame Alignment error. The Boundary Register points to the first packet in the Ring not yet read by the host. If the local DMA address ever reaches the Boundary, reception is aborted. The Boundary Pointer is also used to initialize the Remote DMA for remov-

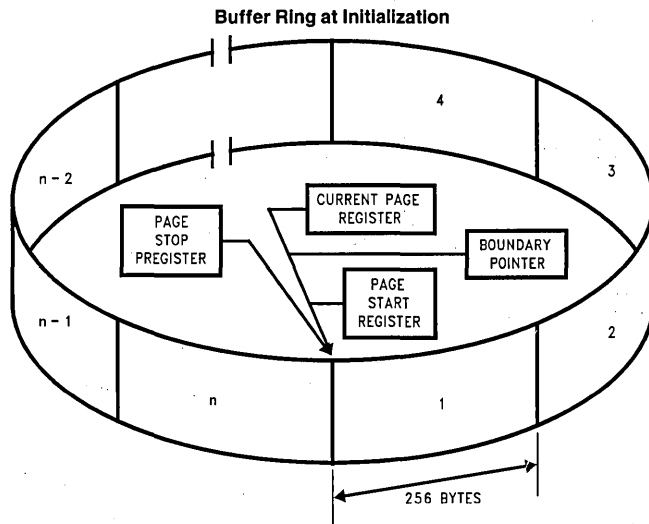
ing a packet and is advanced when a packet is removed. A simple analogy to remember the function of these registers is that the Current Page Register acts as a Write Pointer and the Boundary Pointer acts as a Read Pointer.

**Note:** At initialization, the Page Start Register value should be loaded into both Current Page Register and the Boundary Pointer Register.

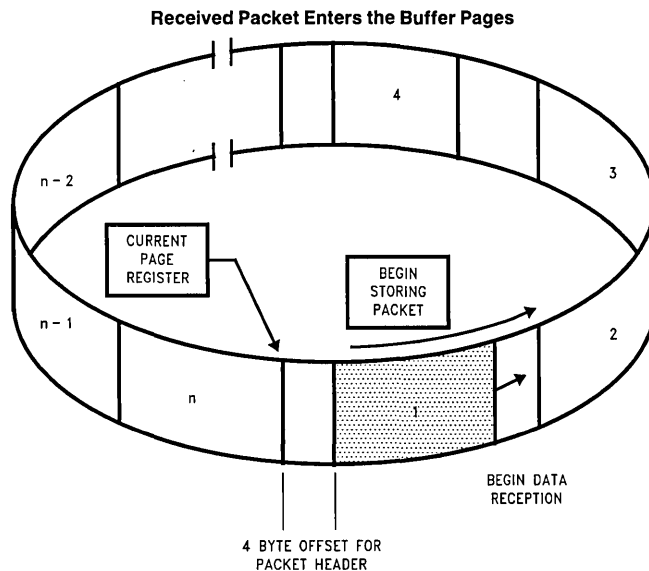
**Note:** The Page Start Register must not be initialized to 00H.

### BEGINNING OF RECEPTION

When the first packet begins arriving the ST-NIC begins storing the packet at the location pointed to by the Current Page Register. An offset of 4 bytes is saved in this first buffer to allow room for storing receive status corresponding to this packet.



TL/F/11157-10



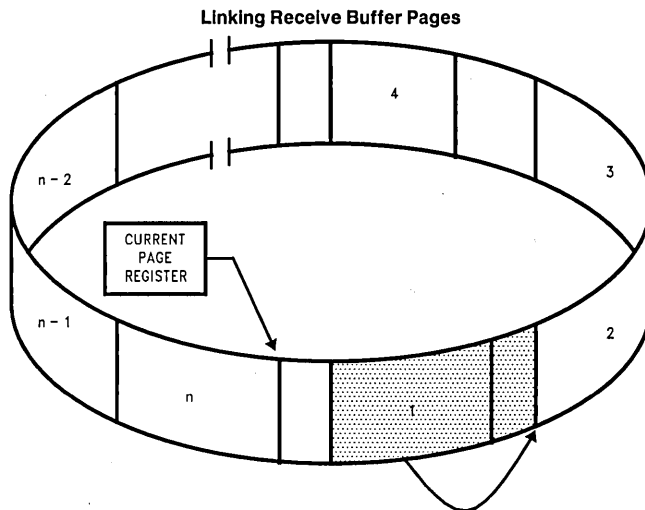
TL/F/11157-11

## 7.0 Packet Reception (Continued)

### LINKING RECEIVE BUFFER PAGES

If the length of the packet exhausts the first 256-byte buffer, the DMA performs a forward link to the next buffer to store the remainder of the packet. For a maximum length packet the buffer logic will link six buffers to store the entire packet. Buffers cannot be skipped when linking, therefore a packet will always be stored in contiguous buffers. Before the next buffer can be linked, the Buffer Management Logic performs two comparisons. The first comparison tests for equality between the DMA address of the next buffer and the contents of the Page Stop Register. If the buffer address equals the Page Stop Register, the buffer management logic will restore the DMA to the first buffer in the

Receive Buffer Ring value programmed in the Page Start Address Register. The second comparison tests for equality between the DMA address of the next buffer address and the contents of the Boundary Pointer Register. If the two values are equal the reception is aborted. The Boundary Pointer Register can be used to protect against overwriting any area in the receive buffer ring that has not yet been read. When linking buffers, buffer management will never cross this pointer, effectively avoiding any overwrites. If the buffer address does not match either the Boundary Pointer or Page Stop Address, the link to the next buffer is performed.



- 1) Check for = to PSTOP
- 2) Check for = to Boundary

TL/F/11157-12

## 7.0 Packet Reception (Continued)

### Buffer Ring Overflow

If the Buffer Ring has been filled and the DMA reaches the Boundary Pointer Address, reception of the incoming packet will be aborted by the ST-NIC. Thus, the packets previously received and still contained in the Ring will not be destroyed.

In heavily loaded network which cause overflows of the Receive Buffer Ring, the ST-NIC may disable the local DMA and suspend further receptions even if the Boundary register is advanced beyond the Current register. To guarantee this will not happen, a software reset must be issued during all Receive Buffer Ring overflows (indicated by the OVW bit in the Interrupt Status Register). The following procedure is required to recover from a Receiver Buffer Ring Overflow.

If this routine is not adhered to, the ST-NIC may act in an unpredictable manner. It should also be noted that it is not permissible to service an overflow interrupt by continuing to empty packets from the receive buffer without implementing the prescribed overflow routine. A flow chart of the ST-NIC's overflow routine follows.

**Note:** It is necessary to define a variable in the driver, which will be called "Resend".

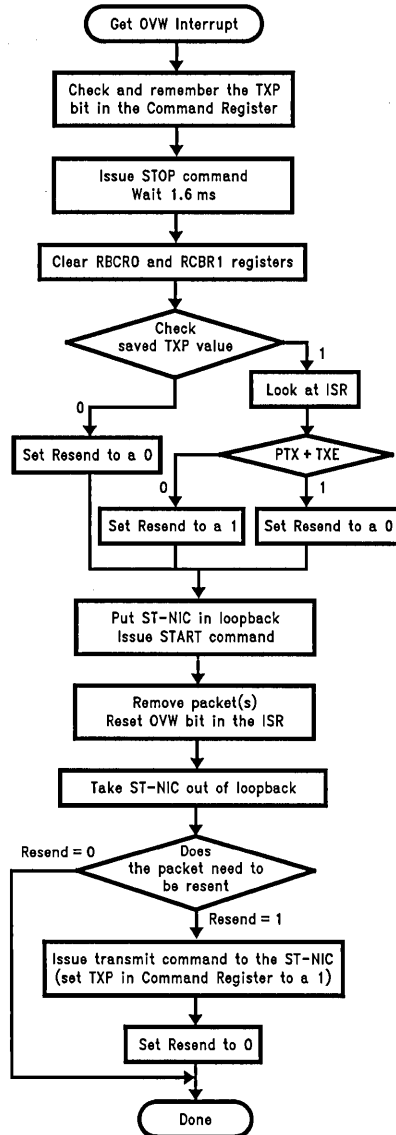
1. Read and store the value of the TXP bit in the ST-NIC Command Register.
2. Issue the STOP command to the ST-NIC. This is accomplished by setting the STP bit in the ST-NIC's Command Register. Writing 21H to the Command Register will stop the ST-NIC.
3. Wait for at least 1.6 ms. Since the ST-NIC will complete any transmission or reception that is in progress, it is necessary to time out for the maximum possible duration of an Ethernet transmission or reception. By waiting 1.6 ms this is achieved with some guard band added. Previously, it was recommended that the RST bit of the Interrupt Status Register be polled to insure that the pending transmission or reception is completed. This bit is not a reliable indicator and subsequently should be ignored.
4. Clear the ST-NIC's Remote Byte Count registers (RBCR0 and RBCR1).
5. Read the stored value of the TXP bit from step 1, above. If this value is a 0, set the "Resend" variable to a 0 and jump to step 6. If this value is a 1, read the ST-NIC's Interrupt Status Register. If either the Packet Transmitted bit (PTX) or Transmit Error bit (TXE) is set to a 1, set the "Resend" variable to a 1 and jump to step 6. If neither of these bits is set, place a 1 in the "Resend" variable and jump to step 6.

If this value is a 1, read the ST-NIC's Interrupt Status Register. If either the Packet Transmitted bit (PTX) or Transmit Error bit (TXE) is set to a 1, set the "Resend" variable to a 1 and jump to step 6. If neither of these bits is set, place a 1 in the "Resend" variable and jump to step 6.

This step determines if there was a transmission in progress when the stop command was issued in step 2. If there was a transmission in progress, the ST-NIC's ISR is read to determine whether or not the packet was recognized by the ST-NIC. If neither the PTX nor TXE bit was set, then the packet will essentially be lost and re-transmitted only after a time-out takes place in the upper level software. By determining that the packet was lost at the driver level, a transmit command can be reissued to

the ST-NIC once the overflow routine is completed (as in step 11). Also, it is possible for the ST-NIC to defer indefinitely, when it is stopped on a busy network. Step 5 also alleviates this problem. Step 5 is essential and should not be omitted from the overflow routine, in order for the ST-NIC to operate correctly.

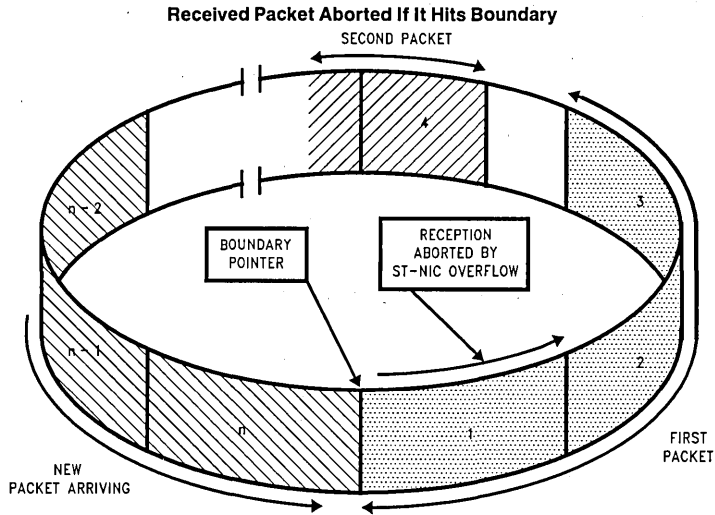
**Overflow Routine Flow Chart**



TL/F/11157-57

## 7.0 Packet Reception (Continued)

6. Place the ST-NIC in either mode 1 or mode 2 loopback. This can be accomplished by setting bits D2 and D1, of the Transmit Configuration Register, to "0,1" or "1,0", respectively.
  7. Issue the START command to the ST-NIC. This can be accomplished by writing 22H to the Command Register. This is necessary to activate the ST-NIC's Remote DMA channel.
  8. Remove one or more packets from the receive buffer ring.
  9. Reset the overwrite warning (OVW, overflow) bit in the Interrupt Status Register.
  10. Take the ST-NIC out of loopback. This is done by writing the Transmit Configuration Register with the value it contains during normal operation. (Bits D2 and D1 should both be programmed to 0.)
  11. If the "Resend" variable is set to a 1, reset the "Resend" variable and reissue the transmit command. This is done by writing a value of 26H to the Command Register. If the "Resend" variable is 0, nothing needs to be done.
- Note 1:** If Remote DMA is not being used, the ST-NIC does not need to be started before packets can be removed from the receive buffer ring. Hence, step 8 could be done before step 7.
- Note 2:** When the ST-NIC is in STOP mode, the Missed Talley Counter is disabled



TL/F/11157-13

## 7.0 Packet Reception (Continued)

### Enabling the ST-NIC On An Active Network

After the ST-NIC has been initialized the procedure for disabling and then re-enabling the ST-NIC on the network is similar to handling Receive Buffer Ring overflow as described previously.

1. Program Command Register for page 0 (Command Register = 21H)
2. Initialize Data Configuration Register (DCR)
3. Clear Remote Byte Count Registers (RBCR0, RBCR1)
4. Initialize Receive Configuration Register (RCR)
5. Place the ST-NIC in LOOPBACK mode 1 or 2 (Transmit Configuration Register = 02H or 04H)
6. Initialize Receive Buffer Ring: Boundary Pointer (BNDRY), Page Start (PSTART), and Page Stop (PSTOP)
7. Clear Interrupt Status Register (ISR) by writing 0FFH to it.
8. Initialize Interrupt Mask Register (IMR)
9. Program Command Register for page 1 (Command Register = 61H)
  - i. Initialize Physical Address Registers (PAR0-PAR5)
  - ii. Initialize Multicast Address Registers (MAR0-MAR7)
  - iii. Initialize CURRENT pointer

10. Put ST-NIC in START mode (Command Register = 22H). The local receive DMA is still not active since the ST-NIC is in LOOPBACK.

11. Initialize the Transmit Configuration for the intended value. The ST-NIC is now ready for transmission and reception.

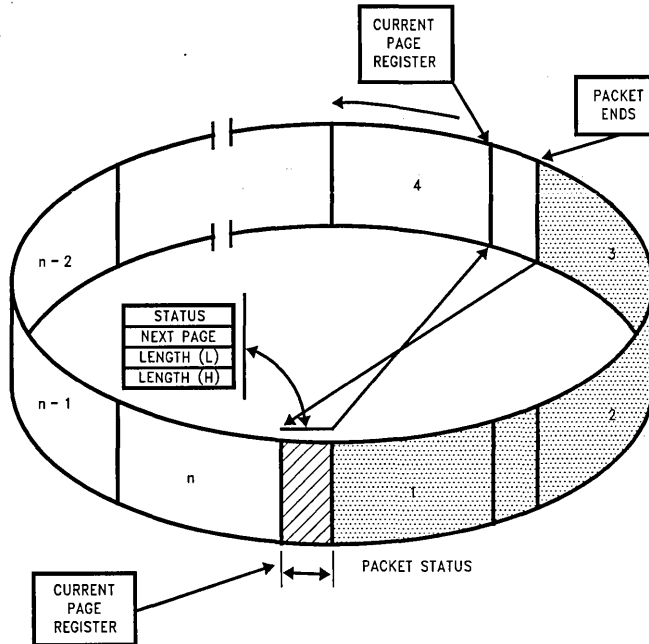
### END OF PACKET OPERATIONS

At the end of the packet the ST-NIC determines whether the received packet is to be accepted or rejected. It either branches to a routine to store the Buffer Header or to another routine that recovers the buffers used to store the packet.

### SUCCESSFUL RECEPTION

If the packet is successfully received, the DMA is restored to the first buffer used to store the packet (pointed to by the Current Page Register). The DMA then stores the Receive Status, a Pointer to where the next packet will be stored (Buffer 4) and the number of received bytes. Note that the remaining bytes in the last buffer are discarded and reception of the next packet begins on the next empty 256-byte buffer boundary. The Current Page Register is then initialized to the next available buffer in the Buffer Ring. (The location of the next buffer had been previously calculated and temporarily stored in an internal scratchpad register.)

Termination of Received Packet—Packet Accepted



TL/F/11157-14

## 7.0 Packet Reception (Continued)

### BUFFER RECOVERY FOR REJECTED PACKETS

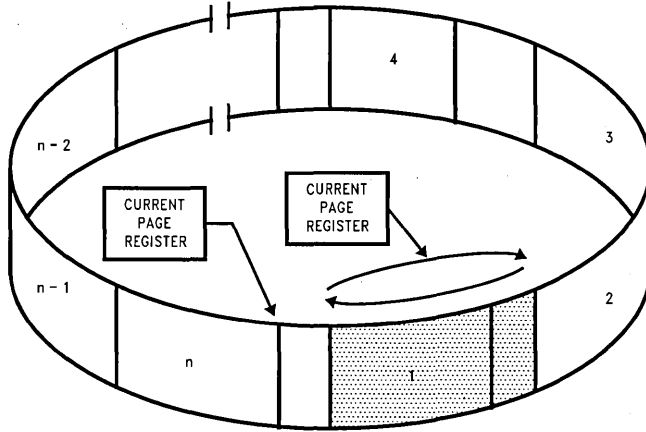
If the packet is a runt packet or contains CRC or Frame Alignment errors, it is rejected. The buffer management logic resets the DMA back to the first buffer page used to store the packet (pointed to by CURR), recovering all buffers that had been used to store the rejected packet. This operation will not be performed if the ST-NIC is programmed to accept either runt packets or packets with CRC or Frame Alignment

errors. The received CRC is always stored in buffer memory after the last byte of received data for the packet.

### Error Recovery

If the packet is rejected as shown, the DMA is restored by the ST-NIC by reprogramming the DMA starting address pointed to by the Current Page Register.

Termination of Receive Packet—Packet Reject



TL/F/11157-15

## 7.0 Packet Reception (Continued)

### REMOVING PACKETS FROM THE RING

Packets are removed from the ring using the Remote DMA or an external device. When using the Remote DMA the Send Packet command can be used. This programs the Remote DMA to automatically remove the received packet pointed to by the Boundary Pointer. At the end of the transfer, the ST-NIC moves the Boundary Pointer, freeing additional buffers for reception. The Boundary Pointer can also be moved manually by programming the Boundary Register.

The ST-NIC knows the difference between an empty buffer ring and a full buffer ring. This situation is seen when the Boundary Pointer (BNDRY) and the Current Page Pointer (CURR) point to the same address. If BNDRY caught up with CURR the buffer is empty and if CURR caught up with BNDRY the buffer is full.

### STORAGE FORMAT FOR RECEIVED PACKETS

The following diagrams describe the format for how received packets are placed into memory by the local DMA channel. These modes are selected in the Data Configuration Register.

AD15	AD8	AD7	AD0
Next Packet Pointer		Receive Status	
Receive Byte Count 0		Receive Byte Count 1	
Byte 1		Byte 2	

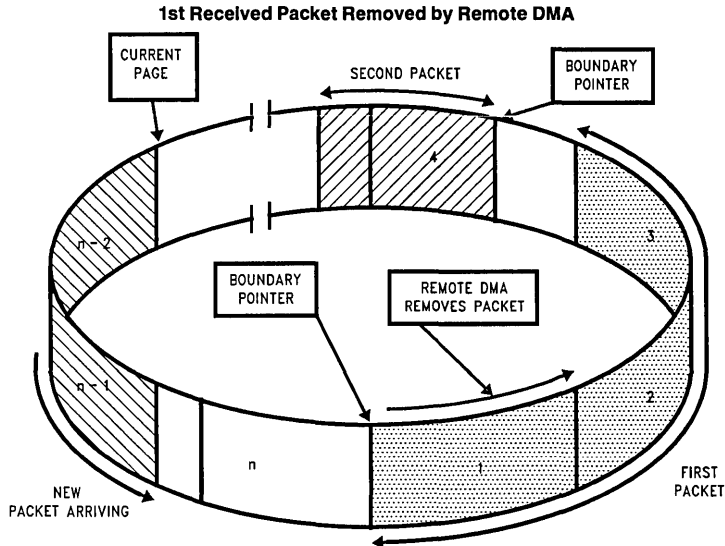
BOS = 1, WTS = 1 in Data Configuration Register. This format is used with 680x0 type processors. (Note: The Receive Count ordering remains the same for BOS = 0 or 1.)

Receive Status
Next Packet Pointer
Receive Byte Count 0
Receive Byte Count 1
Byte 0
Byte 1

BOS = 0, WTS = 0 in Data Configuration Register. This format is used with general 8-bit processors.

AD15	AD8	AD7	AD0
Next Packet Pointer		Receive Status	
Receive Byte Count 1		Receive Byte Count 0	
Byte 2		Byte 1	

BOS = 0, WTS = 1 in Data Configuration Register. This format is used with Series 32xxx, or 680x0 processors.



TL/F/11157-16



## 8.0 Packet Transmission

The Local DMA is also used during transmission of a packet. Three registers control the DMA transfer during transmission, a Transmit Page Start Address Register (TPSR) and the Transmit Byte Count Registers (TBCR0, 1). When the ST-NIC receives a command to transmit the packet pointed to by these registers, buffer memory data will be moved into the FIFO as required during transmission. The ST-NIC will generate and append the preamble, synch and CRC fields.

### General Transmit Packet Format

Transmit	Destination Address	6 Bytes
Byte	Source Address	6 Bytes
Count	Type/Length	2 Bytes
TBCR0, 1	Data Pad (If Data < 46 Bytes)	≥ 46 Bytes

### TRANSMIT PACKET ASSEMBLY

The ST-NIC requires a contiguous assembled packet with the format shown. The transmit byte count includes the Destination Address, Source Address, Length Field and Data. It does not include preamble and CRC. When transmitting data smaller than 46 bytes, the packet must be padded to a minimum size of 64 bytes. The programmer is responsible for adding and stripping pad bytes.

### TRANSMISSION

Prior to transmission, the TPSR (Transmit Page Start Register) and TBCR0, TBCR1 (Transmit Byte Count Registers) must be initialized. To initiate transmission of the packet the TXP bit in the Command Register is set. The Transmit Status Register (TSR) is cleared and the ST-NIC begins to prefetch transmit data from memory (unless the ST-NIC is currently receiving). If the interframe gap has timed out the ST-NIC will begin transmission.

### CONDITIONS REQUIRED TO BEGIN TRANSMISSION

In order to transmit a packet, the following three conditions must be met:

1. The Interframe Gap Timer has timed out the first 6.4  $\mu$ s of the Interframe Gap.
2. At least one byte has entered the FIFO. (This indicates that the burst transfer has been started.)
3. If a collision has been detected the backoff timer has expired.

In typical systems the ST-NIC prefetches the first burst of bytes before the 6.4  $\mu$ s timer expires. The time during which ST-NIC transmits preamble can also be used to load the FIFO.

**Note:** If carrier sense is asserted before a byte has been loaded into the FIFO, the ST-NIC will become a receiver.

### COLLISION RECOVERY

During transmission, the Buffer Management logic monitors the transmit circuitry to determine if a collision has occurred. If a collision is detected, the Buffer Management logic will reset the FIFO and restore the Transmit DMA pointers for retransmission of the packet. The COL bit will be set in the TSR and the NCR (Number of Collisions Register) will be incremented. If 15 retransmissions each result in a collision the transmission will be aborted and the ABT bit in the TSR will be set.

**Note:** NCR reads as zeroes if excessive collisions are encountered.

### TRANSMIT PACKET ASSEMBLY FORMAT

The following diagrams describe the format for how packets must be assembled prior to transmission for different byte ordering schemes. The various formats are selected in the Data Configuration Register.

D15	D8 D7	D0
Destination Address 1	Destination Address 0	
Destination Address 3	Destination Address 2	
Destination Address 5	Destination Address 4	
Source Address 1	Source Address 0	
Source Address 3	Source Address 2	
Source Address 5	Source Address 4	
Type/Length 1	Type/Length 0	
Data 1	Data 0	

BOS = 0, WTS = 1 in Data Configuration Register.

This format is used with Series 32xxx, or 808xx processors.

D15	D8 D7	D0
Destination Address 0	Destination Address 1	
Destination Address 2	Destination Address 3	
Destination Address 4	Destination Address 5	
Source Address 0	Source Address 1	
Source Address 2	Source Address 3	
Source Address 4	Source Address 5	
Type/Length 0	Type/Length 1	
Data 0	Data 1	

BOS = 1, WTS = 1 in Data Configuration Register.

This format is used with 680x0 type processors.

D1	D0
Destination Address 0	
Destination Address 1	
Destination Address 2	
Destination Address 3	
Destination Address 4	
Destination Address 5	
Source Address 0	
Source Address 1	
Source Address 2	
Source Address 3	
Source Address 4	
Source Address 5	

BOS = 0, WTS = 0 in Data Configuration Register.

This format is used with general 8-bit processors.

**Note:** All examples above will result in a transmission of a packet in order of DA0, DA1, DA2, DA3 ... bits within each byte will be transmitted least significant bit first.

DA = Destination Address.

## 9.0 Remote DMA

The Remote DMA channel is used to both assemble packets for transmission, and to remove received packets from the Receive Buffer Ring. It may also be used as a general purpose slave DMA channel for moving blocks of data or commands between host memory and local buffer memory. There are three modes of operation, Remote Write, Remote Read, or Send Packet.

Two register pairs are used to control the Remote DMA, a Remote Start Address (RSAR0, RSAR1) register pair and a Remote Byte Count (RBCR0, RBCR1) register pair. The Start Address Register pair points to the beginning of the block to be moved while the Byte Count Register pair is used to indicate the number of bytes to be transferred. Full handshake logic is provided to move data between local buffer memory and a bidirectional I/O port.

### REMOTE WRITE

A Remote Write transfer is used to move a block of data from the host into local buffer memory. The Remote DMA will read data from the I/O port and sequentially write it to local buffer memory beginning at the Remote Start Address. The DMA Address will be incremented and the Byte Counter will be decremented after each transfer. The DMA is terminated when the Remote Byte Count Register reaches zero.

### REMOTE READ

A Remote Read transfer is used to move a block of data from local buffer memory to the host. The Remote DMA will

sequentially read data from the local buffer memory, beginning at the Remote Start Address, and write data to the I/O port. The DMA Address will be incremented and the Byte Counter will be decremented after each transfer. The DMA is terminated when the Remote Byte Count Register reaches zero.

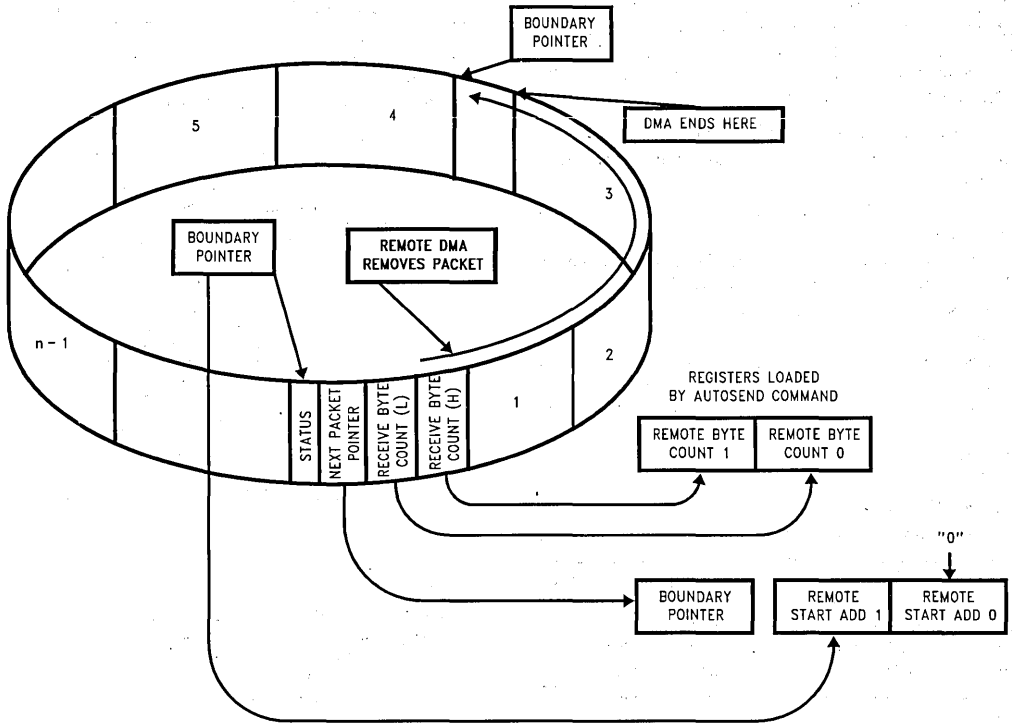
### SEND PACKET COMMAND

The Remote DMA channel can be automatically initialized to transfer a single packet from the Receive Buffer Ring. The CPU begins this transfer by issuing a "Send Packet" Command. The DMA will be initialized to the value of the Boundary Pointer Register and the Remote Byte Count Register pair (RBCR0, RBCR1) will be initialized to the value of the Receive Byte Count fields found in the Buffer Header of each packet. After the data is transferred, the Boundary Pointer is advanced to allow the buffers to be used for new receive packets. The Remote Read will terminate when the Byte Count equals zero. The Remote DMA is then prepared to read the next packet from the Receive Buffer Ring. If the DMA pointer crosses the Page Stop Register, it is reset to the Page Start Address. This allows the Remote DMA to remove packets that have wrapped around to the top of the Receive Buffer Ring.

**Note 1:** In order for the ST-NIC to correctly execute the Send Packet Command, the upper Remote Byte Count Register (RBCR1) must first be loaded with 0FH.

**Note 2:** The Send Packet command cannot be used with 680x0 type processors.

Remote DMA Autoinitialization from Buffer Ring

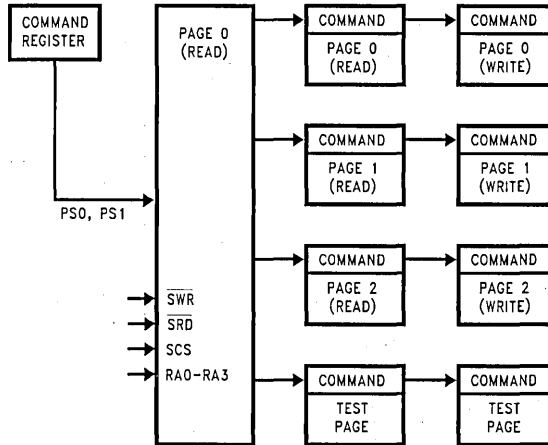


## 10.0 Internal Registers

All registers are 8-bit wide and mapped into four pages which are selected in the Command Register (PS0, PS1). Pins RA0-RA3 are used to address registers within each page. Page 0 registers are those registers which are com-

monly accessed during ST-NIC operation while page 1 registers are used primarily for initialization. The registers are partitioned to avoid having to perform two write/read cycles to access commonly used registers.

### 10.1 REGISTER ADDRESS MAPPING



TL/F/11157-18

### 10.2 REGISTER ADDRESS ASSIGNMENTS

Page 0 Address Assignments (PS1 = 0, PS0 = 0)

RA0-RA3	RD	WR
00H	Command (CR)	Command (CR)
01H	Current Local DMA Address 0 (CLDA0)	Page Start Register (PSTART)
02H	Current Local DMA Address 1 (CLDA1)	Page Stop Register (PSTOP)
03H	Boundary Pointer (BNRY)	Boundary Pointer (BNRY)
04H	Transmit Status Register (TSR)	Transmit Page Start Address (TPSR)
05H	Number of Collisions Register (NCR)	Transmit Byte Count Register 0 (TBCR0)
06H	FIFO (FIFO)	Transmit Byte Count Register 1 (TBCR1)
07H	Interrupt Status Register (ISR)	Interrupt Status Register (ISR)
08H	Current Remote DMA Address 0 (CRDA0)	Remote Start Address Register 0 (RSAR0)

RA0-RA3	RD	WR
09H	Current Remote DMA Address 1 (CRDA1)	Remote Start Address Register 1 (RSAR1)
0AH	Reserved	Remote Byte Count Register 0 (RBCR0)
0BH	Reserved	Remote Byte Count Register 1 (RBCR1)
0CH	Receive Status Register (RSR)	Receive Configuration Register (RCR)
0DH	Tally Counter 0 (Frame Alignment Errors) (CNTR0)	Transmit Configuration Register (TCR)
0EH	Tally Counter 1 (CRC Errors) (CNTR1)	Data Configuration Register (DCR)
0FH	Tally Counter 2 Missed Packet Errors) (CNTR2)	Interrupt Mask Register (IMR)

## 10.0 Internal Registers (Continued)

Page 1 Address Assignments (PS1 = 0, PS0 = 1)

RA0-RA3	RD	WR
00H	Command (CR)	Command (CR)
01H	Physical Address Register 0 (PAR0)	Physical Address Register 0 (PAR0)
02H	Physical Address Register 1 (PAR1)	Physical Address Register 1 (PAR1)
03H	Physical Address Register 2 (PAR2)	Physical Address Register 2 (PAR2)
04H	Physical Address Register 3 (PAR3)	Physical Address Register 3 (PAR3)
05H	Physical Address Register 4 (PAR4)	Physical Address Register 4 (PAR4)
06H	Physical Address Register 5 (PAR5)	Physical Address Register 5 (PAR5)
07H	Current Page Register (CURR)	Current Page Register (CURR)
08H	Multicast Address Register 0 (MAR0)	Multicast Address Register 0 (MAR0)
09H	Multicast Address Register 1 (MAR1)	Multicast Address Register 1 (MAR1)
0AH	Multicast Address Register 2 (MAR2)	Multicast Address Register 2 (MAR2)
0BH	Multicast Address Register 3 (MAR3)	Multicast Address Register 3 (MAR3)
0CH	Multicast Address Register 4 (MAR4)	Multicast Address Register 4 (MAR4)
0DH	Multicast Address Register 5 (MAR5)	Multicast Address Register 5 (MAR5)
0EH	Multicast Address Register 6 (MAR6)	Multicast Address Register 6 (MAR6)
0FH	Multicast Address Register 7 (MAR7)	Multicast Address Register 7 (MAR7)

Page 2 Address Assignments (PS1 = 1, PS0 = 0)

RA0-RA3	RD	WR
00H	Command (CR)	Command (CR)
01H	Page Start Register (PSTART)	Current Local DMA Address 0 (CLDA0)
02H	Page Stop Register (PSTOP)	Current Local DMA Address 1 (CLDA1)
03H	Remote Next Packet Pointer	Remote Next Packet Pointer
04H	Transmit Page Start Address (TPSR)	Reserved
05H	Local Next Packet Pointer	Local Next Packet Pointer
06H	Address Counter (Upper)	Address Counter (Upper)
07H	Address Counter (Lower)	Address Counter (Lower)
08H	Reserved	Reserved
09H	Reserved	Reserved
0AH	Reserved	Reserved
0BH	Reserved	Reserved
0CH	Receive Configuration Register (RCR)	Reserved
0DH	Transmit Configuration Register (TCR)	Reserved
0EH	Data Configuration Register (DCR)	Reserved
0FH	Interrupt Mask Register (IMR)	Reserved

**Note:** Page 2 registers should only be accessed for diagnostic purposes. They should not be modified during normal operation.

Page 3 should never be modified.

## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS

#### COMMAND REGISTER (CR) 00H (READ/WRITE)

The Command Register is used to initiate transmissions, enable or disable Remote DMA operations and to select register pages. To issue a command the microprocessor sets the corresponding bit(s) (RD2, RD1, RD0, TXP). Further commands may be overlapped, but with the following rules: (1) If a transmit command overlaps with a remote DMA operation, bits RD0, RD1, and RD2 must be maintained for the remote DMA command when setting the TXP bit. Note, if a remote DMA command is re-issued when giving the transmit command, the DMA will complete immediately if the remote byte count register has not been reinitialized. (2) If a remote DMA operation overlaps a transmission, RD0, RD1, and RD2 may be written with the desired values and a "0" written to the TXP bit. Writing a "0" to this bit has no effect. (3) A remote write DMA may not overlap remote read operation or vice versa. Either of these operations must either complete or be aborted before the other operation may start. Bits PS1, PS0, RD2, and STP may be set any time.

7	6	5	4	3	2	1	0
PS1	PS0	RD2	RD1	RD0	TXP	STA	STP

Bit	Symbol	Description																								
D0	STP	<p><b>Stop:</b> Software reset command, takes the controller offline, no packets will be received or transmitted. Any reception or transmission in progress will continue to completion before entering the reset state. To exit this state, the STP bit must be reset and the STA bit must be set high. To perform a software reset, this bit should be set high. The software reset has executed only when indicated by the RST bit in the ISR being set to 1. <b>STP powers up high.</b></p> <p><b>Note:</b> If the ST-NIC has previously been in start mode and the STP is set, both the STP and STA bits will remain set.</p>																								
D1	STA	<p><b>Start:</b> This bit is used to activate the ST-NIC after either power up, or when the ST-NIC has been placed in a reset mode by software command or error. <b>STA powers up low.</b></p>																								
D2	TXP	<p><b>Transmit Packet:</b> This bit must be set to initiate the transmission of a packet. TXP is internally reset either after the transmission is completed or aborted. This bit should be set only after the Transmit Byte Count and Transmit Page Start registers have been programmed.</p>																								
D3, D4, and D5	RD0, RD1, and RD2	<p><b>Remote DMA Command:</b> These three encoded bits control operation of the Remote DMA channel. RD2 can be set to abort any Remote DMA command in progress. The Remote Byte Count Registers should be cleared when a Remote DMA has been aborted. The Remote Start Addresses are not restored to the starting address if the Remote DMA is aborted.</p> <table style="margin-left: 20px;"> <tr> <td style="text-align: center;">RD2</td> <td style="text-align: center;">RD1</td> <td style="text-align: center;">RD0</td> <td></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>Not Allowed</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>Remote Read</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>Remote Write (Note 2)</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Send Packet</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">X</td> <td style="text-align: center;">X</td> <td>Abort/Complete Remote DMA (Note 1)</td> </tr> </table> <p><b>Note 1:</b> If a remote DMA operation is aborted and the remote byte count has not decremented to zero, PRQ will remain high. A read acknowledge (RACK) on a write acknowledge (WACK) will reset PRQ low.</p> <p><b>Note 2:</b> For proper operation of the Remote Write DMA, there are two steps which must be performed before using the Remote Write DMA. The steps are as follows:</p> <ol style="list-style-type: none"> <li>I) Write a non-zero value into RBCR0.</li> <li>II) Set bits RD2, RD1, and RD0 to 0, 0, and 1.</li> <li>III) Set RBCR0, 1 and RSAR0, 1.</li> <li>IV) Issue the Remote Write DMA Command (RD2, RD1, RD0 = 0, 1, 0).</li> </ol>	RD2	RD1	RD0		0	0	0	Not Allowed	0	0	1	Remote Read	0	1	0	Remote Write (Note 2)	0	1	1	Send Packet	1	X	X	Abort/Complete Remote DMA (Note 1)
RD2	RD1	RD0																								
0	0	0	Not Allowed																							
0	0	1	Remote Read																							
0	1	0	Remote Write (Note 2)																							
0	1	1	Send Packet																							
1	X	X	Abort/Complete Remote DMA (Note 1)																							
D6 and D7	PS0 and PS1	<p><b>Page Select:</b> These two encoded bits select which register page is to be accessed with addresses RA0-3.</p> <table style="margin-left: 20px;"> <tr> <td style="text-align: center;">PS1</td> <td style="text-align: center;">PS0</td> <td></td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>Register Page 0</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>Register Page 1</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>Register Page 2</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>Reserved</td> </tr> </table>	PS1	PS0		0	0	Register Page 0	0	1	Register Page 1	1	0	Register Page 2	1	1	Reserved									
PS1	PS0																									
0	0	Register Page 0																								
0	1	Register Page 1																								
1	0	Register Page 2																								
1	1	Reserved																								



## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS (Continued)

#### INTERRUPT STATUS REGISTER (ISR) 07H (READ/WRITE)

This register is accessed by the host processor to determine the cause of an interrupt. Any interrupt can be masked in the Interrupt Mask Register (IMR). Individual interrupt bits are cleared by writing a "1" into the corresponding bit of the ISR. The INT signal is active as long as any unmasked signal is set, and will not go low until all unmasked bits in this register have been cleared. The ISR must be cleared after power up by writing it with all 1's.

7	6	5	4	3	2	1	0
RST	RDC	CNT	OVW	TXE	RXE	PTX	PRX

Bit	Symbol	Description
D0	PRX	<b>Packet Received:</b> Indicates packet received with no errors.
D1	PTX	<b>Packet Transmitted:</b> Indicates packet transmitted with no errors.
D2	RXE	<b>Receive Error:</b> Indicates that a packet was received with one or more of the following errors: — CRC Error — Frame Alignment Error — FIFO Overrun — Missed Packet
D3	TXE	<b>Transmit Error:</b> Set when packet transmitted with one or more of the following errors: — Excessive Collisions — FIFO Underrun
D4	OVW	<b>Overwrite Warning:</b> Set when receive buffer ring storage resources have been exhausted. (Local DMA has reached Boundary Pointer)
D5	CNT	<b>Counter Overflow:</b> Set when MSB of one or more of the Network Tally Counters has been set.
D6	RDC	<b>Remote DMA Complete:</b> Set when Remote DMA operation has been completed.
D7	RST	<b>Reset Status:</b> Set when ST-NIC enters reset state and cleared when a Start Command is issued to the CR. This bit is also set when a Receive Buffer Ring overflow occurs and is cleared when one or more packets have been removed from the ring. Writing to this bit has no effect.  <b>Note:</b> This bit does not generate an interrupt, it is merely a status indicator.

## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS (Continued)

#### INTERRUPT MASK REGISTER (IMR) 0FH (WRITE)

The Interrupt Mask Register is used to mask interrupts. Each interrupt mask bit corresponds to a bit in the Interrupt Status Register (ISR). If an interrupt mask bit is set, an interrupt will be issued whenever the corresponding bit in the ISR is set. If any bit in the IMR is set low, an interrupt will not occur when the bit in the ISR is set. **The IMR powers up to all zeroes.**

7	6	5	4	3	2	1	0
—	RDCE	CNTE	OVWE	TXEE	RXEE	PTXE	PRXE

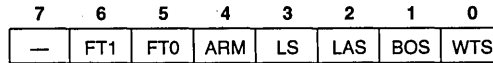
Bit	Symbol	Description
D0	PRXE	<b>Packet Received Interrupt Enable</b> 0: Interrupt Disabled 1: Enables Interrupt when packet received
D1	PTXE	<b>Packet Transmitted Interrupt Enable</b> 0: Interrupt Disabled 1: Enables Interrupt when packet is transmitted
D2	RXEE	<b>Receive Error Interrupt Enable</b> 0: Interrupt Disabled 1: Enables Interrupt when packet received with error
D3	TXEE	<b>Transmit Error Interrupt Enable</b> 0: Interrupt Disabled 1: Enables Interrupt when packet transmission results in error
D4	OVWE	<b>Overwrite Warning Interrupt Enable</b> 0: Interrupt Disabled 1: Enables Interrupt when Buffer Management Logic lacks sufficient buffers to store incoming packet
D5	CNTE	<b>Counter Overflow Interrupt Enable</b> 0: Interrupt Disabled 1: Enables Interrupt when MSB of one or more of the Network Statistics counters has been set
D6	RDCE	<b>DMA Complete Interrupt Enable</b> 0: Interrupt Disabled 1: Enables Interrupt when Remote DMA transfer has been completed
D7	Reserved	Reserved

## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS (Continued)

#### DATA CONFIGURATION REGISTER (DCR) 0EH (WRITE)

This Register is used to program the ST-NIC for 8- or 16-bit memory interface, select byte ordering in 16-bit applications and establish FIFO thresholds. **The DCR must be initialized prior to loading the Remote Byte Count Registers. LAS is set on power up.**



Bit	Symbol	Description																				
D0	WTS	<p><b>Word Transfer Select</b>                      0: Selects byte-wide DMA transfers                      1: Selects word-wide DMA transfers</p> <p>; WTS establishes byte or word transfers for both Remote and Local DMA transfers  <b>Note:</b> When word-wide mode is selected up to 32k words are addressable; A0 remains low.</p>																				
D1	BOS	<p><b>Byte Order Select</b>                      0: MS byte placed on AD15–AD8 and LS byte on AD7–AD0. (32xxx, 80x86)                      1: MS byte placed on AD7–AD0 and LS byte on AD15–A8. (680x0)</p> <p>; Ignored when WTS is low</p>																				
D2	LAS	<p><b>Long Address Select</b>                      0: Dual 16-bit DMA mode                      1: Single 32-bit DMA mode</p> <p>; When LAS is high, the contents of the Remote DMA registers RSAR0, 1 are issued as A16–A31 Power up high</p>																				
D3	LS	<p><b>Loopback Select</b>                      0: Loopback mode selected. Bits D1 and D2 of the TCR must also be programmed for Loopback operation                      1: Normal Operation</p>																				
D4	ARM	<p><b>Auto-Initialize Remote</b>                      0: Send Command not executed, all packets removed from Buffer Ring under program control                      1: Send Command executed, Remote DMA auto-initialized to remove packets from Buffer Ring  <b>Note:</b> Send Command cannot be used with 680x0 byte processors.</p>																				
D5 and D6	FT0 and FT1	<p><b>FIFO Threshold Select:</b> Encoded FIFO threshold. Establishes point at which bus is requested when filling or emptying the FIFO. During reception, the FIFO threshold indicates the number of bytes (or words) the FIFO has filled serially from the network before bus request (BREQ) is asserted.</p> <p><b>Note:</b> FIFO threshold setting determines the DMA burst length.</p> <p style="text-align: center;">Receive Thresholds</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th style="text-align: center;">FT1</th> <th style="text-align: center;">FT0</th> <th style="text-align: center;">Word Wide</th> <th style="text-align: center;">Byte Wide</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1 Word</td> <td style="text-align: center;">2 Bytes</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">2 Words</td> <td style="text-align: center;">4 Bytes</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">4 Words</td> <td style="text-align: center;">8 Bytes</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">6 Words</td> <td style="text-align: center;">12 Bytes</td> </tr> </tbody> </table> <p>During transmission, the FIFO threshold indicates the number of bytes (or words) the FIFO has filled from the Local DMA before BREQ is asserted. Thus, the transmission threshold is 13 bytes minus the received threshold.</p>	FT1	FT0	Word Wide	Byte Wide	0	0	1 Word	2 Bytes	0	1	2 Words	4 Bytes	1	0	4 Words	8 Bytes	1	1	6 Words	12 Bytes
FT1	FT0	Word Wide	Byte Wide																			
0	0	1 Word	2 Bytes																			
0	1	2 Words	4 Bytes																			
1	0	4 Words	8 Bytes																			
1	1	6 Words	12 Bytes																			



## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS (Continued)

#### TRANSMIT CONFIGURATION REGISTER (TCR) 0DH (WRITE)

The transmit configuration establishes the actions of the transmitter section of the ST-NIC during transmission of a packet on the network. **LB1 and LB0 which select loopback mode power up as 0.**

7	6	5	4	3	2	1	0
—	—	—	OFST	ATD	LB1	LB0	CRC

Bit	Symbol	Description																				
D0	CRC	<p><b>Inhibit CRC</b></p> <p>0: CRC appended by transmitter                      1: CRC inhibited by transmitter</p> <p>In loopback mode CRC can be enabled or disabled to test the CRC logic</p>																				
D1 and D2	LB0 and LB1	<p><b>Encoded Loopback Control:</b> These encoded configuration bits set the type of loopback that is to be performed. Note that loopback in mode 2 places the ENDEC Module in loopback mode and that D3 of the DCR must be set to zero for loopback operation.</p> <table style="margin-left: 40px; border-collapse: collapse;"> <tr> <td></td> <td style="text-align: center;">LB1</td> <td style="text-align: center;">LB0</td> <td></td> </tr> <tr> <td>Mode 0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>Normal Operation (LPBK = 0)</td> </tr> <tr> <td>Mode 1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>Internal NIC Module Loopback (LPBK = 0)</td> </tr> <tr> <td>Mode 2</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>Internal ENDEC Module Loopback (LPBK = 1)</td> </tr> <tr> <td>Mode 3</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>External Loopback (LPBK = 0)</td> </tr> </table>		LB1	LB0		Mode 0	0	0	Normal Operation (LPBK = 0)	Mode 1	0	1	Internal NIC Module Loopback (LPBK = 0)	Mode 2	1	0	Internal ENDEC Module Loopback (LPBK = 1)	Mode 3	1	1	External Loopback (LPBK = 0)
	LB1	LB0																				
Mode 0	0	0	Normal Operation (LPBK = 0)																			
Mode 1	0	1	Internal NIC Module Loopback (LPBK = 0)																			
Mode 2	1	0	Internal ENDEC Module Loopback (LPBK = 1)																			
Mode 3	1	1	External Loopback (LPBK = 0)																			
D3	ATD	<p><b>Auto Transmit Disable:</b> This bit allows another station to disable the ST-NIC's transmitter by transmission of a particular multicast packet. The transmitter can be re-enabled by resetting this bit or by reception of a second particular multicast packet.</p> <p>1: Reception of multicast address hashing to bit 62 disables transmitter, reception of multicast address hashing to bit 63 enables transmitter.</p>																				
D4	OFST	<p><b>Collision Offset Enable:</b> This bit modifies the backoff algorithm to allow prioritization of nodes.</p> <p>0: Backoff Logic implements normal algorithm.                      1: Forces Backoff algorithm modification to 0 to <math>2^{\min(3+n, 10)}</math> slot times for first three collisions, then follows standard backoff. (For the first three collisions, the station has higher average backoff delay making a low priority mode.)</p>																				
D5	Reserved	Reserved																				
D6	Reserved	Reserved																				
D7	Reserved	Reserved																				

## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS (Continued)

#### TRANSMIT STATUS REGISTER (TSR) 04H (READ)

This register records events that occur on the media during transmission of a packet. It is cleared when the next transmission is initiated by the host. All bits remain low unless the event that corresponds to a particular bit occurs during transmission. Each transmission should be followed by a read of this register. The contents of this register are not specified until after the first transmission.

7	6	5	4	3	2	1	0
OWC	CDH	FU	CRS	ABT	COL	—	PTX

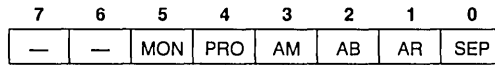
Bit	Symbol	Description
D0	PTX	<b>Packet Transmitted:</b> Indicates transmission without error. (No excessive collisions or FIFO underrun) (ABT = "0", FU = "0")
D1	Reserved	Reserved
D2	COL	<b>Transmit Collided:</b> Indicates that the transmission collided at least once with another station on the network. The number of collisions is recorded in the Number of Collisions Registers (NCR).
D3	ABT	<b>Transmit Aborted:</b> Indicates the ST-NIC aborted transmission because of excessive collisions. (Total number of transmissions including original transmission attempt equals 16.)
D4	CRS	<b>Carrier Sense Lost:</b> This bit is set when carrier is lost during transmission of the packet. Transmission is not aborted on loss of carrier.
D5	FU	<b>FIFO Underrun:</b> If the ST-NIC cannot gain access of the bus before the FIFO empties, this bit is set. Transmission of the packet will be aborted.
D6	CDH	<b>CD Heartbeat:</b> Failure of the transceiver to transmit a collision signal after transmission of a packet will set this bit. The Collision Detect (CD) heartbeat signal must commence during the first 6.4 $\mu$ s of the Interframe Gap following a transmission. In certain collisions, the CD Heartbeat bit will be set even though the transceiver is not performing the CD heartbeat test.
D7	OWC	<b>Out of Window Collision:</b> Indicates that a collision occurred after a slot time (51.2 $\mu$ s). Transmissions rescheduled as in normal collisions.

## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS (Continued)

#### RECEIVE CONFIGURATION REGISTER (RCR) OCH (WRITE)

This register determines operation of the ST-NIC during reception of a packet and is used to program what types of packets to accept.



Bit	Symbol	Description
D0	SEP	<p><b>Save Errored Packets</b></p> <p>0: Packets with receive errors are rejected.                      1: Packets with receive errors are accepted. Receive errors are CRC and Frame Alignment errors.</p>
D1	AR	<p><b>Accept Runt Packets:</b> This bit allows the receiver to accept packets that are smaller than 64 bytes. The packet must be at least 8 bytes long to be accepted as a runt.</p> <p>0: Packets with fewer than 64 bytes rejected.                      1: Packets with fewer than 64 bytes accepted.</p>
D2	AB	<p><b>Accept Broadcast:</b> Enables the receiver to accept a packet with an all 1's destination address.</p> <p>0: Packets with broadcast destination address rejected.                      1: Packets with broadcast destination address accepted.</p>
D3	AM	<p><b>Accept Multicast:</b> Enables the receiver to accept a packet with a multicast address. All multicast addresses must pass the hashing array.</p> <p>0: Packets with multicast destination address not checked.                      1: Packets with multicast destination address checked.</p>
D4	PRO	<p><b>Promiscuous Physical:</b> Enables the receiver to accept all packets with a physical address.</p> <p>0: Physical address of node must match the station address programmed in PAR0–PAR5.                      1: All packets with physical addresses accepted.</p>
D5	MON	<p><b>Monitor Mode:</b> Enables the receiver to check addresses and CRC on incoming packets without buffering to memory. The Missed Packet Tally counter will be incremented for each recognized packet.</p> <p>0: Packets buffered to memory.                      1: Packets checked for address match, good CRC and Frame Alignment but not buffered to memory.</p>
D6	Reserved	Reserved
D7	Reserved	Reserved

**Note:** D2 and D3 are "OR'd" together, i.e., if D2 and D3 are set the ST-NIC will accept broadcast and multicast addresses as well as its own physical address. To establish full promiscuous mode, bits D2, D3, and D4 should be set. In addition the multicast hashing array must be set to all 1's in order to accept all multicast addresses.



## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS (Continued)

#### RECEIVE STATUS REGISTER (RSR) OCH (READ)

This register records status of the received packet, including information on errors and the type of address match, either physical or multicast. The contents of this register are written to buffer memory by the DMA after reception of a good packet. If packets with errors are to be saved the receive status is written to memory at the head of the erroneous packet if an erroneous packet is received. If packets with errors are to be rejected the RSR will not be written to memory. The contents will be cleared when the next packet arrives. CRC errors, Frame Alignment errors and missed packets are counted internally by the ST-NIC which relinquishes the Host from reading the RSR in real time to record errors for Network Management Functions. The contents of this register are not specified until after the first reception.

7	6	5	4	3	2	1	0
DFR	DIS	PHY	MPA	FO	FAE	CRC	PRX

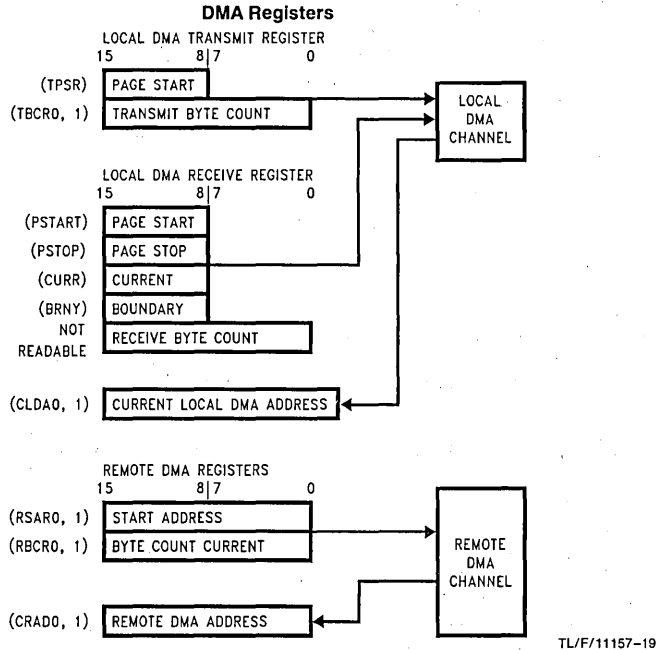
Bit	Symbol	Description
D0	PRX	<b>Packet Received Intact:</b> Indicates packet received without error. (Bits CRC, FAE, FO, and MPA are zero for the received packet.)
D1	CRC	<b>CRC Error:</b> Indicates packet received with CRC error. Increments Tally Counter (CNTR1). This bit will also be set for Frame Alignment errors.
D2	FAE	<b>Frame Alignment Error:</b> Indicates that the incoming packet did not end on a byte boundary and the CRC did not match at the last byte boundary. Increments Tally Counter (CNTR0).
D3	FO	<b>FIFO Overrun:</b> This bit is set when the FIFO is not serviced causing overflow during reception. Reception of the packet will be aborted.
D4	MPA	<b>Missed Packet:</b> Set when a packet intended for node cannot be accepted by ST-NIC because of a lack of receive buffers or if the controller is in monitor mode and did not buffer the packet to memory. Increments Tally Counter (CNTR2).
D5	PHY	<b>Physical/Multicast Address:</b> Indicates whether received packet had a physical or multicast address type. 0: Physical Address Match 1: Multicast/Physical Address Match
D6	DIS	<b>Receiver Disabled:</b> Set when receiver disabled by entering Monitor mode. Reset when receiver is re-enabled when exiting Monitor mode.
D7	DFR	<b>Deferring:</b> Set when internal Carrier Sense or Collision signals are generated in the ENDEC module. If the transceiver has asserted the CD line as a result of the jabber, this bit will stay set indicating the jabber condition.

**Note:** Following coding applies to CRC and FAE bits.

FAE	CRC	Type of Error
0	0	No Error (Good CRC and <6 Dribble Bits)
0	1	CRC Error
1	0	Illegal, Will Not Occur
1	1	Frame Alignment Error and CRC Error

# 10.0 Internal Registers (Continued)

## 10.4 DMA REGISTERS



The DMA Registers are partitioned into groups; Transmit, Receive and Remote DMA Registers. The Transmit registers are used to initialize the Local DMA Channel for transmission of packets while the Receive Registers are used to initialize the Local DMA Channel for packet Reception. The Page Stop, Page Start, Current and Boundary Registers are used by the Buffer Management Logic to supervise the Receive Buffer Ring. The Remote DMA Registers are used to initialize the Remote DMA.

**Note:** In the figure above, registers are shown as 8 or 16 bits wide. Although some registers are 16-bit internal registers, all registers are accessed as 8-bit registers. Thus the 16-bit Transmit Byte Count Register is broken into two 8-bit registers, TBCR0, TBCR1. Also TPSR, PSTART, PSTOP, CURR and BRNY only check or control the upper 8 bits of address information on the bus. Thus, they are shifted to positions 15-8 in the diagram above.

### 10.5 TRANSMIT DMA REGISTERS

#### TRANSMIT PAGE START REGISTER (TPSR)

This register points to the assembled packet to be transmitted. Only the eight higher order addresses are specified since all transmit packets are assembled on 256-byte page boundaries. The bit assignment is shown below. The values placed in bits D7-D0 will be used to initialize the higher order address (A8-A15) of the Local DMA for transmission. The lower order bits (A7-A0) are initialized to zero.

#### Bit Assignment

	7	6	5	4	3	2	1	0
TPSR	A15	A14	A13	A12	A11	A10	A9	A8

(A7-A0 Initialized to Zero)

#### TRANSMIT BYTE COUNT REGISTER 0, 1 (TBCR0, TBCR1)

These two registers indicate the length of the packet to be transmitted in bytes. The count must include the number of bytes in the source, destination, length and data fields. The maximum number of transmit bytes allowed is 64 Kbytes. The ST-NIC will not truncate transmissions longer than 1500 bytes. The bit assignment is shown below:

	7	6	5	4	3	2	1	0
TBCR1	L15	L14	L13	L12	L11	L10	L9	L8

	7	6	5	4	3	2	1	0
TBCR0	L7	L6	L5	L4	L3	L2	L1	L0

## 10.0 Internal Registers (Continued)

### 10.6 LOCAL DMA RECEIVE REGISTERS

#### PAGE START AND STOP REGISTERS (PSTART, PSTOP)

The Page Start and Page Stop Registers program the starting and stopping address of the Receive Buffer Ring. Since the ST-NIC uses fixed 256-byte buffers aligned on page boundaries only the upper 8 bits of the start and stop address are specified.

#### PSTART, PSTOP Bit Assignment

	7	6	5	4	3	2	1	0
PSTART, PSTOP	A15	A14	A13	A12	A11	A10	A9	A8

#### BOUNDARY (BNRY) REGISTER

This register is used to prevent overflow of the Receive Buffer Ring. Buffer management compares the contents of this register to the next buffer address when linking buffers together. If the contents of this register match the next buffer address the Local DMA operation is aborted.

	7	6	5	4	3	2	1	0
BNRY	A15	A14	A13	A12	A11	A10	A9	A8

#### CURRENT PAGE REGISTER (CURR)

This register is used internally by the Buffer Management Logic as a backup register for reception. CURR contains the address of the first buffer to be used for a packet reception and is used to restore DMA pointers in the event of receive errors. This register is initialized to the same value as PSTART and should not be written to again unless the controller is Reset.

	7	6	5	4	3	2	1	0
CURR	A15	A14	A13	A12	A11	A10	A9	A8

#### CURRENT LOCAL DMA REGISTER 0,1 (CLDA0, 1)

These two registers can be accessed to determine the current local DMA address.

	7	6	5	4	3	2	1	0
CLDA1	A15	A14	A13	A12	A11	A10	A9	A8

	7	6	5	4	3	2	1	0
CLDA0	A7	A6	A5	A4	A3	A2	A1	A0

### 10.7 REMOTE DMA REGISTERS

#### REMOTE START ADDRESS REGISTERS (RSAR0, 1)

Remote DMA operations are programmed via the Remote Start Address (RSAR0, 1) and Remote Byte Count (RBCR0, 1) registers. The Remote Start Address is used to point to the start of the block of data to be transferred and the Remote Byte Count is used to indicate the length of the block (in bytes).

	7	6	5	4	3	2	1	0
RSAR1	A15	A14	A13	A12	A11	A10	A9	A8

	7	6	5	4	3	2	1	0
RSAR0	A7	A6	A5	A4	A3	A2	A1	A0

#### REMOTE BYTE COUNT REGISTERS (RBCR0, 1)

	7	6	5	4	3	2	1	0
RBCR1	BC15	BC14	BC13	BC12	BC11	BC10	BC9	BC8

	7	6	5	4	3	2	1	0
RBCR0	BC7	BC6	BC5	BC4	BC3	BC2	BC1	BC0

**Note:** RSAR0 programs the start address bits A0-A7.

RSAR1 programs the start address bits A8-A15.

Address incremented by two for word transfers, and by one for byte transfers. Byte Count decremented by two for word transfers and by one for byte transfers.

RBCR0 programs LSB byte count.

RBCR1 programs MSB byte count.

#### CURRENT REMOTE DMA ADDRESS (CRDA0, CRDA1)

The Current Remote DMA Registers contain the current address of the Remote DMA. The bit assignment is shown below:

	7	6	5	4	3	2	1	0
CRDA1	A15	A14	A13	A12	A11	A10	A9	A8

	7	6	5	4	3	2	1	0
CRDA0	A7	A6	A5	A4	A3	A2	A1	A0

### 10.8 PHYSICAL ADDRESS REGISTERS (PAR0-PAR5)

The physical address registers are used to compare the destination address of incoming packets for rejecting or accepting packets. Comparisons are performed on a byte-wide basis. The bit assignment shown below relates the sequence in PAR0-PAR5 to the bit sequence of the received packet.

	D7	D6	D5	D4	D3	D2	D1	D0
PAR0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
PAR1	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
PAR2	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
PAR3	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
PAR4	DA39	DA38	DA37	DA36	DA35	DA34	DA33	DA32
PAR5	DA47	DA46	DA45	DA44	DA43	DA42	DA41	DA40

	Destination Address						Source		
P/S	DA0	DA1	DA2	DA3	...	DA46	DA47	SA0	...

**Note:** P/S = Preamble, Sync

DA0 = Physical/Multicast Bit

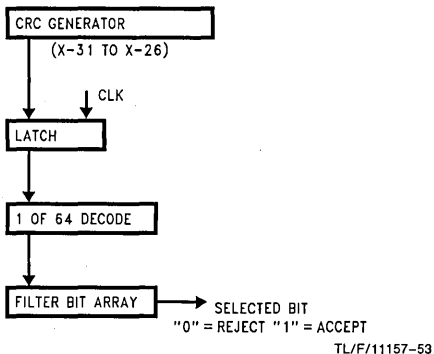
### 10.9 MULTICAST ADDRESS REGISTERS (MAR0-MAR7)

The multicast address registers provide filtering of multicast addresses hashed by the CRC logic. All destination addresses are fed through the CRC logic and as the last bit of the destination address enters the CRC, the 6 most signifi-

## 10.0 Internal Registers (Continued)

cant bits of the CRC generator are latched. These 6 bits are then decoded by a 1 of 64 decode to index a unique filter bit (FB0-63) in the multicast address registers. If the filter bit selected is set, the multicast packet is accepted. The system designer would use a program to determine which filter bits to set in the multicast registers. All multicast filter bits that correspond to multicast address accepted by the node are then set to one. To accept all multicast packets all of the registers are set to all ones.

**Note:** Although the hashing algorithm does not guarantee perfect filtering of multicast address, it will perfectly filter up to 64 multicast addresses if these addresses are chosen to map into unique locations in the multicast filter.



	D7	D6	D5	D4	D3	D2	D1	D0
MAR0	FB7	FB6	FB5	FB4	FB3	FB2	FB1	FB0
MAR1	FB15	FB14	FB13	FB12	FB11	FB10	FB9	FB8
MAR2	FB23	FB22	FB21	FB20	FB19	FB18	FB17	FB16
MAR3	FB31	FB30	FB29	FB28	FB27	FB26	FB25	FB24
MAR4	FB39	FB38	FB37	FB36	FB35	FB34	FB33	FB32
MAR5	FB47	FB46	FB45	FB44	FB43	FB42	FB41	FB40
MAR6	FB55	FB54	FB53	FB52	FB51	FB50	FB49	FB48
MAR7	FB63	FB62	FB61	FB60	FB59	FB58	FB57	FB56

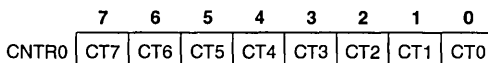
If address Y is found to hash to the value 32 (20H), then FB32 in MAR4 should be initialized to "1". This will cause the ST-NIC to accept any multicast packet with the address Y.

### 10.10 NETWORK TALLY COUNTERS

Three 8-bit counters are provided for monitoring the number of CRC errors, Frame Alignment Errors and Missed Packets. The maximum count reached by any counter is 192 (COH). These registers will be cleared when read by the CPU. The count is recorded in binary in CT0-CT7 of each Tally Register.

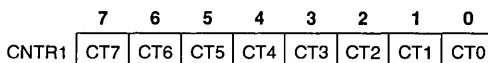
#### Frame Alignment Error Tally (CNTR0)

This counter increments every time a packet is received with a Frame Alignment Error. The packet must have been recognized by the address recognition logic. The counter is cleared after it is read by the processor.



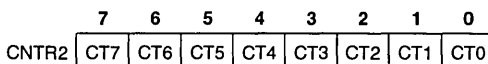
#### CRC Error Tally (CNTR1)

This counter is incremented every time a packet is received with a CRC error. The packet must first be recognized by the address recognition logic. The counter is cleared after it is read by the processor.



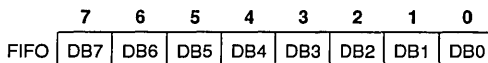
#### Frames Lost Tally Register (CNTR2)

This counter is incremented if a packet cannot be received due to lack of buffer resources. In monitor mode, this counter will count the number of packets that pass the address recognition logic.



#### FIFO

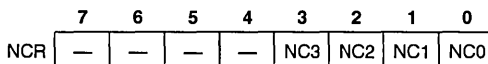
This is an 8-bit register that allows the CPU to examine the contents of the FIFO after loopback. The FIFO will contain the last 8 data bytes transmitted in the loopback packet. Sequential reads from the FIFO will advance a pointer in the FIFO and allow reading of all 8 bytes.



**Note:** The FIFO should only be read when the ST-NIC has been programmed in loopback mode.

#### NUMBER OF COLLISIONS (NCR)

This register contains the number of collisions a node experiences when attempting to transmit a packet. If no collisions are experienced during a transmission attempt, the COL bit of the TSR will not be set and the contents of NCR will be zero. If there are excessive collisions, the ABT bit in the TSR will be set and the contents of NCR will be zero. The NCR is cleared after the TXP bit in the CR is set.



## 11.0 Initialization Procedures

The ST-NIC must be initialized prior to transmission or reception of packets from the network. Power on reset is applied to the ST-NIC's reset pin. This clears/sets the following bits:

Register	Reset Bits	Set Bits
Command Register (CR)	TXP, STA	RD2, STP
Interrupt Status (ISR)		RST
Interrupt Mask (IMR)	All Bits	
Data Control (DCR)		LAS
Transmit Config. (TCR)	LB1, LB0	

The ST-NIC remains in its reset state until a Start Command is issued. This guarantees that no packets are transmitted or received and that the ST-NIC remains a bus slave until all appropriate internal registers have been programmed. After initialization the STP bit of the command register is reset and packets may be received and transmitted.

### Initialization Sequence

The following initialization procedure is mandatory.

1. Program Command Register for Page 0 (Command Register = 21H)
2. Initialize Data Configuration Register (DCR)
3. Clear Remote Byte Count Registers (RBCR0, RBCR1)
4. Initialize Receive Configuration Register (RCR)
5. Place the ST-NIC in LOOPBACK mode 1 or 2 (Transmit Configuration Register = 02H or 04H)
6. Initialize Receive Buffer Ring: Boundary Pointer (BNDRY), Page Start (PSTART), and Page Stop (PSTOP)
7. Clear Interrupt Status Register (ISR) by writing 0FFH to it.
8. Initialize Interrupt Mask Register (IMR)
9. Program Command Register for page 1 (Command Register = 61H)
  - I) Initialize Physical Address Registers (PAR0-PAR5)
  - II) Initialize Multicast Address Registers (MAR0-MAR5)
  - III) Initialize CURRent pointer
10. Put ST-NIC in START mode (Command Register = 22H).
11. Initialize the Transmit Configuration Register for the intended value. The ST-NIC is now ready for transmission and reception.

Before receiving packets, the user must specify the location of the Receive Buffer Ring. This is programmed in the Page Start and Page Stop Registers. In addition, the Boundary and Current Page Register must be initialized to the value of the Page Start Register. These registers will be modified during reception of packets.

## 12.0 Loopback Diagnostics

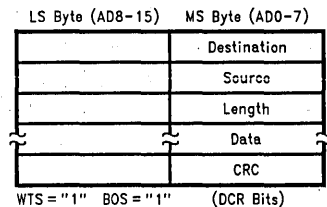
Three forms of local loopback are provided on the ST-NIC. The user has the ability to loopback through the deserializer on the controller, through the ENDEC module or the Transceiver. **Because of the half duplex architecture of the ST-NIC, loopback testing is a special mode of operation with the following restrictions:**

### Restrictions During Loopback

The FIFO is split into two halves, one half is used for transmission and the other for reception. Only 8-bit fields can be fetched from memory so two tests are required for 16-bit systems to verify integrity of the entire data path. During loopback the maximum latency from the assertion of BREQ to BACK is 2.0  $\mu$ s. Systems that wish to use the loopback test but do not meet this latency can limit the loopback to 7 bytes without experiencing underflow. Only the last 8 bytes of the loopback packet are retained in the FIFO. The last 8 bytes can be read through the FIFO register which will advance through the FIFO to allow reading the receive packet sequentially.

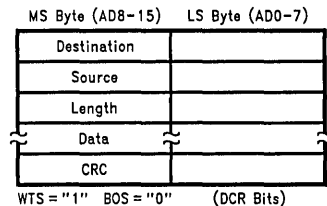
Destination Address	= (6 Bytes) Station Physical Address
Source Address	
Length	2 Bytes
Data	= 46 to 1500 Bytes
CRC	Appended by ST-NIC if CRC = "0" in TCR

When in word-wide mode with Byte Order Select set, the loopback packet must be assembled in the even byte location as shown below. (Loopback only operates with byte wide transfers.)



TL/F/11157-54

When in word-wide mode with Byte Order Select low, the following format must be used for the loopback packet.



TL/F/11157-55

**Note:** When using loopback in word mode 2n bytes must be programmed in TBCR0, 1. Where n = actual number of bytes assembled in even or odd location.



## 12.0 Loopback Diagnostics (Continued)

To initiate a loopback the user first assembles the loopback packet then selects the type of loopback using the Transmit Configuration register bits LB0, LB1. The transmit configuration register must also be set to enable or disable CRC generation during transmission. The user then issues a normal transmit command to send the packet. During loopback the receiver checks for an address match and if CRC bit in the TCR is set, the receiver will also check the CRC. The last 8 bytes of the loopback packet are buffered and can read out of the FIFO using the FIFO read port.

### Loopback Modes

**MODE 1:** Loopback through the NIC Module (LB1 = 0, LB0 = 1): If this loopback is used, the NIC Modules's serial-izer is connected to the deserializer.

**MODE 2:** Loopback through the ENDEC Module (LB1 = 1, LB0 = 0): If the loopback is to be performed through the SNI, the ST-NIC provides a control (LPBK) that forces the ENDEC module to loopback all signals.

**MODE 3:** Loopback to cable (LB1 = 1, LB0 = 1). Packets can be transmitted to the cable in loopback mode to check all of the transmit and receive paths and the cable itself.

**Note:** Collision and Carrier Sense can be generated by the ENDEC module and are masked by the NIC module. It is not possible to go directly between the loopback modes, it is necessary to return to normal operation (00H) when changing modes.

### Reading the Loopback Packet

The last 8 bytes of a received packet can be examined by 8 consecutive reads of the FIFO register. The FIFO pointer is incremented after the rising edge of the CPU's read strobe by internally synchronizing and advancing the pointer. This may take up to four bus clock cycles, if the pointer has not been incremented by the time the CPU reads the FIFO register again, the ST-NIC will insert wait states.

**Note:** The FIFO may only be read during Loopback. Reading the FIFO at any other time will cause the ST-NIC to malfunction.

### Alignment of the Received Packet in the FIFO

Reception of the packet in the FIFO begins at location zero, after the FIFO pointer reaches the last location in the FIFO, the pointer wraps to the top of the FIFO overwriting the previously received data. This process is continued until the last byte is received. The ST-NIC then appends the received byte count in the next two locations of the FIFO. The contents of the Upper Byte Count are also copied to the next FIFO location. The number of bytes used in the loopback packet determines the alignment of the packet in the FIFO.

The alignment for a 64-byte packet is shown below.

FIFO Location	FIFO Contents	
0	Lower Byte Count	First Byte Read
1	Upper Byte Count	Second Byte Read
2	Upper Byte Count	•
3	Last Byte	•
4	CRC1	•
5	CRC2	•
6	CRC3	•
7	CRC4	Last Byte Read

For the following alignment in the FIFO the packet length should be  $(N \times 8) + 5$  Bytes. Note that if the CRC bit in the TCR is set, CRC will not be appended by the transmitter. If the CRC is appended by the transmitter, the 1st four bytes, bytes N-3 to N, correspond to the CRC.

FIFO Location	FIFO Contents	
0	Byte N-4	First Byte Read
1	Byte N-3 (CRC1)	Second Byte Read
2	Byte N-2 (CRC2)	•
3	Byte N-1 (CRC3)	•
4	Byte N (CRC4)	•
5	Lower Byte Count	•
6	Upper Byte Count	Last Byte Read
7	Upper Byte Count	

### LOOPBACK TESTS

Loopback capabilities are provided to allow certain tests to be performed to validate operation of the DP83902A ST-NIC prior to transmitting and receiving packets on a live network. Typically these tests may be performed during power up of a node. The diagnostic provides support to verify the following:

1. Verify integrity of data path. Received data is checked against transmitted data.
2. Verify the CRC logic's capability to generate good CRC on transmit, verify CRC on receive (good or bad CRC).

## 12.0 Loopback Diagnostics (Continued)

3. Verify that the Address Recognition Logic can

- a) Recognize address match packets
- b) Reject packets that fail to match an address

### LOOPBACK OPERATION IN THE ST-NIC

Loopback is a modified form of transmission using only half of the FIFO. This places certain restrictions on the use of loopback testing. When loopback mode is selected in the TCR, the FIFO is split. A packet should be assembled in memory with programming of TPSR and TBCR0, TBCR1 registers. When the transmit command is issued the following operations occur:

#### Transmitter Actions

1. Data is transferred from memory by the DMA until the FIFO is filled. For each transfer TBCR0 and TBCR1 are decremented. (Subsequent burst transfers are initiated when the number of bytes in the FIFO drops below the programmed threshold.)
2. The ST-NIC generates 56 bits of preamble followed by an 8-bit synch pattern.
3. Data transferred from FIFO to serializer.
4. If CRC = 1 in TCR, the CRC is not calculated by ST-NIC, and the last byte transmitted is the last byte from the FIFO (Allows software CRC to be appended). If CRC = 0, ST-NIC calculates and appends four bytes of CRC.
5. At end of Transmission PTX bit set in ISR.

#### Receiver Actions

1. Wait for synch, all preamble stripped.
2. Store packet in FIFO, increment receive byte count for each incoming byte.
3. If CRC = 1 in TCR, receiver checks incoming packet for CRC errors. If CRC = 0 in TCR, receiver does not check CRC errors, CRC error bit always set in RSR (for address matching packets).
4. At the end of receive, the receive byte count is written into the FIFO, and the receive status register is updated. The PRX bit is typically set in the RSR even if the address does not match. If CRC errors are forced, the packet must match the address filters in order for the CRC error bit in the RSR to be set.

#### EXAMPLES

The following examples show what results can be expected from a properly operating ST-NIC during loopback. The restrictions and results of each type of loopback are listed for reference. The loopback tests are divided into two sets of tests. One to verify the data path, CRC generation and byte count through all three paths. The second set of tests uses internal loopback to verify the receiver's CRC checking and address recognition. For all of the tests the DCR was programmed to 40H.

Path	TCR	RCR	TSR	RSR	ISR
ST-NIC Internal	02	1F	53 (Note 1)	02 (Note 2)	02 (Note 3)

**Note 1:** Since carrier sense and collision detect are generated in the ENDEC module, they are blocked during NIC loopback. Carrier and CD heartbeat are not seen and the CRS and CDH bits are set.

**Note 2:** CRC errors are always indicated by the receiver if CRC is appended by the transmitter.

**Note 3:** Only the PTX bit in the ISR is set, the PRX bit is only set if status is written to memory. In loopback this action does not occur and the PRX bit remains 0 for all loopback modes.

**Note 4:** All values are hex.

Path	TCR	RCR	TSR	RSR	ISR
ST-NIC Internal	04	1F	43 (Note 1)	02	02

**Note 1:** CDH is set, CRS is not set since it is generated by the external encoder/decoder.

Path	TCR	RCR	TSR	RSR	ISR
ST-NIC External	06	1F	03 (Note 1)	02	02 (Note 2)

**Note 1:** CDH and CRS should not be set. The TSR however, could also contain 01H, 03H, 07H and a variety of other values depending on whether collisions were encountered or the packet was deferred.

**Note 2:** The ISR will contain 08H if packet is not transmittable.

**Note 3:** During external loopback the ST-NIC is now exposed to network traffic. It is therefore possible for the contents of both the Receive portion of the FIFO and the RSR to be corrupted by any other packet on the network. Thus in a live network the contents of the FIFO and RSR should not be depended on. The ST-NIC will still abide by the standard CSMA/CD protocol in external loopback mode. (i.e., The network will not be disturbed by the loopback packet.)

**Note 4:** All values are hex.

### CRC AND ADDRESS RECOGNITION

The next three tests exercise the address recognition logic and CRC. These tests should be performed using internal loopback only so that the ST-NIC is isolated from interference from the network. These tests also require the capability to generate CRC in software.

The address recognition logic cannot be directly tested. The CRC and FAE bits in the RSR are only set if the address in the packet matches the address filters. If errors are expected to be set and they are not set, the packet has been rejected on the basis of an address mismatch. The following sequence of packets will test the address recognition logic. The DCR should be set to 40H and the TCR should be set to 03H with a software generated CRC.

Packet Contents			Results
Test	Address	CRC	RSR
Test A	Matching	Good	01 (Note 1)
Test B	Matching	Bad	02 (Note 2)
Test C	Non-Matching	Bad	01

**Note 1:** Status will read 21H if multicast address used.

**Note 2:** Status will read 22H if multicast address used.

**Note 3:** In test A, the RSR is set up. In test B the address is found to match since the CRC is flagged as bad. Test C proves that the address recognition logic can distinguish a bad address and does not notify the RSR of the bad CRC. The receiving CRC is proven to work in test A and test B.

**Note 4:** All values are hex.

### NETWORK MANAGEMENT FUNCTIONS

Network management capabilities are required for maintenance and planning of a local area network. The ST-NIC supports the minimum requirement for network management in hardware, the remaining requirements can be met with software counts. There are three events that software alone can not track during reception of packets: CRC errors, Frame Alignment errors, and missed packets.

### 12.0 Loopback Diagnostics (Continued)

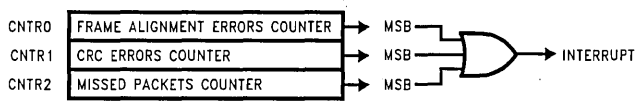
Since errored packets can be rejected, the status associated with these packets is lost unless the CPU can access the Receive Status Register before the next packer arrives. In situations where another packet arrives very quickly, the CPU may have no opportunity to do this. The ST-NIC counts the number of packets with CRC errors and Frame Alignment errors. 8-Bit counters have been selected to reduce overhead. The counters will generate interrupts whenever their MSBs are set so that a software routine can accumulate the network statistics and reset the counter before overflow occurs. The counters are sticky so that when they reach a count of 192 (C0H) counting is halted. An additional counter is provided to count the number of packets the ST-NIC misses due to buffer overflow or being offline.

The structure of the counters is shown below:

Additional information required for network management is available in the Receive and Transmit Status Registers. Transmit status is available after each transmission for information regarding events during transmission.

Typically, the following statistics might be gathered in software:

- Traffic: Frames Sent OK
- Frames Received OK
- Multicast Frames Received
- Packets Lost Due to Lack of Resources
- Retries/Packet
  
- Errors: CRC Errors
- Alignment Errors
- Excessive Collisions
- Packet with Length Errors
- Heartbeat Failure

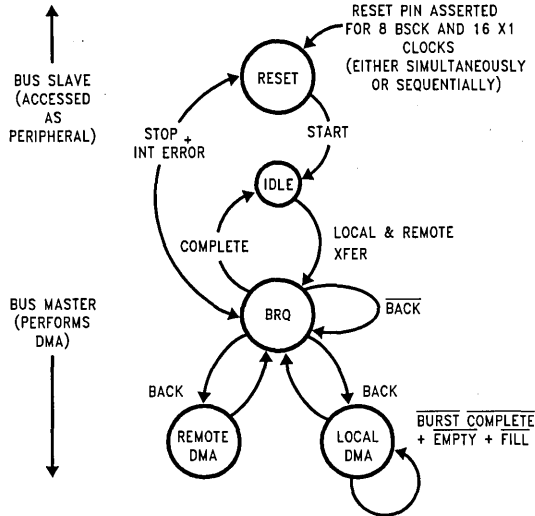


TL/F/11157-20

### 13.0 Bus Arbitration and Timing

The ST-NIC operates in three possible modes:

- BUS MASTER (WHILE PERFORMING DMA)
- BUS SLAVE (WHILE BEING ACCESSED BY CPU)
- IDLE



TL/F/11157-21

Upon power-up the ST-NIC is in an indeterminate state. After receiving a hardware reset the ST-NIC is a bus slave in the Reset State, the receiver and transmitter are both disabled in this state. The reset state can be re-entered under three conditions, soft reset (Stop Command), hard reset (RESET input) or an error that shuts down the receiver of transmitter (FIFO underflow or overflow). After initialization of registers, the ST-NIC is issued a Start command and the ST-NIC enters Idle state. Until the DMA is required the ST-NIC remains in idle state. The idle state is exited by a request from the FIFO on the case of receiver or transmit, or from the Remote DMA in the case of Remote DMA

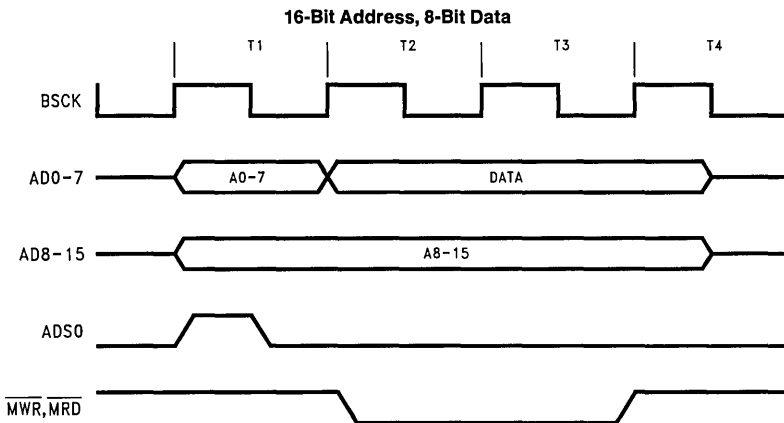
operation. After acquiring the bus in a BREQ/BACK handshake the Remote or Local DMA transfer is completed and the ST-NIC re-enters the idle state.

#### DMA TRANSFERS TIMING

The DMA can be programmed for the following types of transfers:

- 16-Bit Address, 8-bit Data Transfer
- 16-Bit Address, 16-bit Data Transfer
- 32-Bit Address, 8-bit Data Transfer
- 32-Bit Address, 16-bit Data Transfer

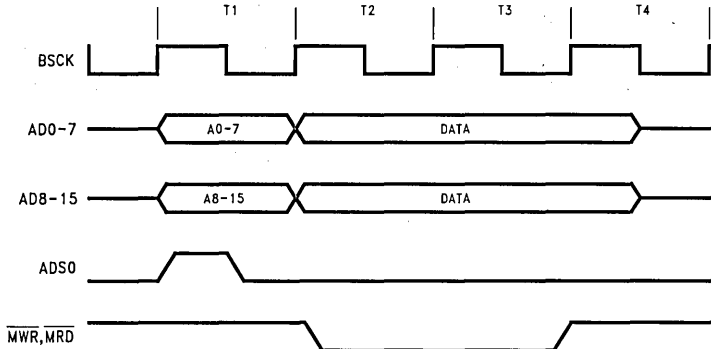
All DMA transfers use BSCK for timing. 16-Bit Address modes require 4 BSCK cycles as shown below:



TL/F/11157-22

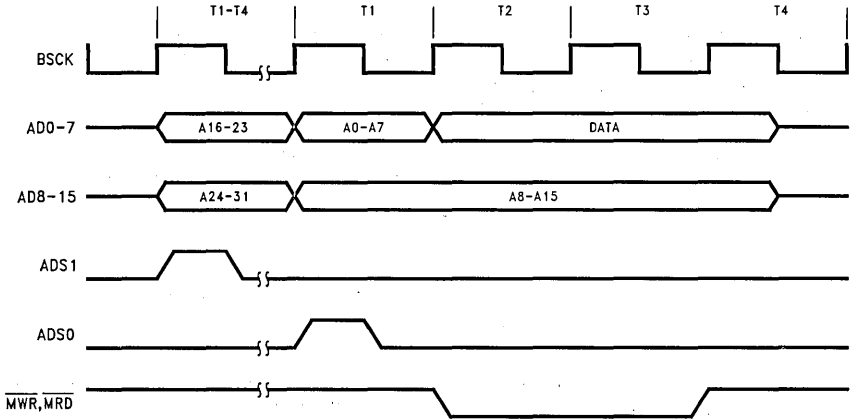
### 13.0 Bus Arbitration and Timing (Continued)

16-Bit Address, 16-Bit Data



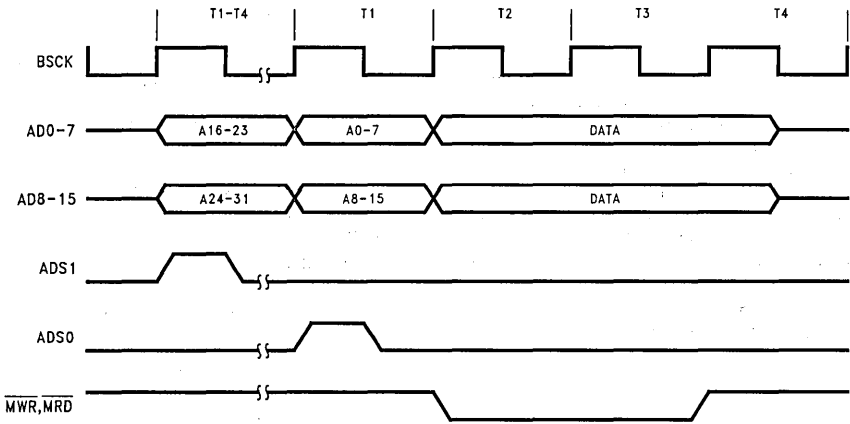
TL/F/11157-23

32-Bit Address, 8-Bit Data



TL/F/11157-24

32-Bit Address, 16-Bit Data



TL/F/11157-25

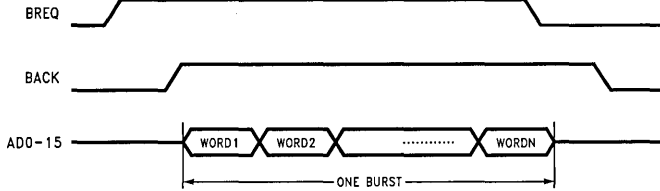
Note: In 32-bit address mode, ADS1 is at TRI-STATE after the first T1-T4 states; thus, a 4.7k pull-down resistor is required for 32-bit address.

### 13.0 Bus Arbitration and Timing (Continued)

When in 32-bit mode four additional BSKC cycles are required per burst. The first bus cycle (T1'-T4') of each burst is used to output the upper 16-bit addresses. This 16-bit address is programmed in RSAR0 and RSAR1 and points to a 64k page of system memory. All transmitted or received packets are constrained to reside within this 64k page.

#### FIFO BURST CONTROL

All Local DMA transfers are burst transfers, once the DMA requests the bus and the bus is acknowledged, the DMA will transfer an exact burst of bytes programmed in the Data Configuration Register (DCR) then relinquish the bus. If there are remaining bytes in the FIFO the next burst will not be initiated until the FIFO threshold is exceeded. If BACK is removed during the transfer, the burst transfer will be aborted. **(DROPPING BACK DURING A DMA CYCLE IS NOT RECOMMENDED.)**



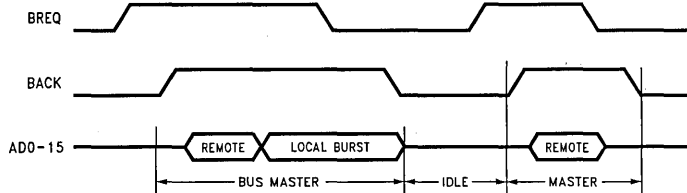
TL/F/11157-26

where N = 1, 2, 4, or 6 Words or N = 2, 4, 8, or 12 Bytes when in byte mode.

#### INTERLEAVED LOCAL OPERATION

If a remote DMA transfer is initiated or in progress when a packet is being received or transmitted, the Remote DMA transfer will be interrupted for higher priority Local DMA

transfers. When the Local DMA transfer is completed the Remote DMA will re-arbitrate for the bus and continue its transfers. This is illustrated below:



TL/F/11157-27

Note that if the FIFO requires service while a remote DMA is in progress, BREQ is not dropped and the Local DMA burst is appended to the Remote Transfer. When switching from a local transfer to a remote transfer, however, BREQ is dropped and raised again. This allows the CPU or other devices to fairly contend for the bus.

#### FIFO AND BUS OPERATIONS

##### Overview

To accommodate the different rates at which data comes from (or goes to) the network and goes to (or comes from) the system memory, the ST-NIC contains a 16-byte FIFO for buffering data between the bus and the media. The FIFO threshold is programmable, allowing filling (or emptying) the FIFO at different rates. When the FIFO has filled to its programmed threshold, the local DMA channel transfers these bytes (or words) into local memory. It is crucial that the local DMA is given access to the bus within a minimum bus latency time; otherwise a FIFO underrun (or overrun) occurs.

To understand FIFO underruns or overruns, there are two causes which produce this condition—

1. the bus latency is so long that the FIFO has filled (or emptied) from the network before the local DMA has serviced the FIFO.
2. the bus latency or bus data rate has slowed the throughput of the local DMA to a point where it is slower than the network data rate (10 Mb/s). This second condition is also dependent upon DMA clock and word width (byte wide or word wide).

The worst case condition ultimately limits the overall bus latency which the ST-NIC can tolerate.

#### FIFO Underrun and Transmit Enable

During transmission, if a FIFO underrun occurs, the Transmit enable (TXE) output may remain high (active). Generally, this will cause a very large packet to be transmitted onto the network. The jabber feature of the transceiver will terminate the transmission, and reset TXE.

To prevent this problem, a properly designed system will not allow FIFO underruns by giving the ST-NIC a bus acknowledge within time shown in the maximum bus latency curves shown and described later.

#### FIFO at the Beginning of Receive

At the beginning of reception, the ST-NIC stores entire Address field of each incoming packet in the FIFO to deter-

### 13.0 Bus Arbitration and Timing (Continued)

mine whether the packet matches its Physical Address Registers or maps to one of its Multicast Registers. This causes the FIFO to accumulate 8 bytes. Furthermore, there are some synchronization delays in the DMA PLA. Thus, the actual time that BREQ is asserted from the time the Start of Frame Delimiter (SFD) is detected is 7.8  $\mu$ s. This operation affects the bus latencies at 2- and 4-byte thresholds during the first receive BREQ since the FIFO must be filled to 8 bytes (or 4 words) before issuing a BREQ.

#### FIFO Operation at the End of Receive

When Carrier Sense goes low, the ST-NIC enters its end of packet processing sequence, emptying its FIFO and writing the status information at the beginning of the packet, *Figure 5*. The ST-NIC holds onto the bus for the entire sequence. The longest time BREQ may be extended occurs when a packet ends just as the ST-NIC performs its last FIFO burst. The ST-NIC, in this case, performs a programmed burst transfer followed by flushing the remaining bytes in the FIFO, and completes by writing the header information to memory. The following steps occur during this sequence.

1. ST-NIC issues BREQ because the FIFO threshold has been reached
2. During the burst, packet ends, resulting in BREQ extended.
3. ST-NIC flushes remaining bytes from FIFO
4. ST-NIC performs internal processing to prepare for writing the header.
5. ST-NIC writes 4-byte (2-word) header
6. ST-NIC deasserts BREQ

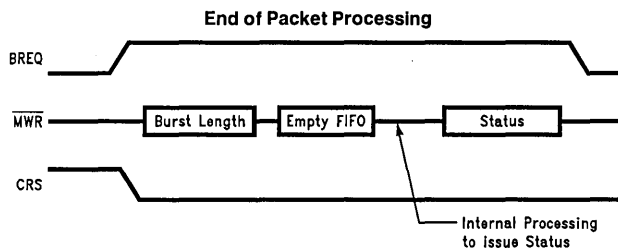
End of Packet Processing (EOPP) times for 10 MHz and 20 MHz have been tabulated in the table below.

Mode	Threshold	Bus Clock	EOPP
Byte	2 Bytes	10 MHz	7.0 $\mu$ s
	4 Bytes		8.6 $\mu$ s
	8 Bytes		11.0 $\mu$ s
Byte	2 Bytes	20 MHz	3.6 $\mu$ s
	4 Bytes		4.2 $\mu$ s
	8 Bytes		5.0 $\mu$ s
Word	2 Bytes	10 MHz	5.4 $\mu$ s
	4 Bytes		6.2 $\mu$ s
	8 Bytes		7.4 $\mu$ s
Word	2 Bytes	20 MHz	3.0 $\mu$ s
	4 Bytes		3.2 $\mu$ s
	8 Bytes		3.6 $\mu$ s

**End of Packet Processing Times for Various FIFO Thresholds, Bus Clocks and Transfer Modes**

#### Threshold Detection (Bus Latency)

To assure that no overwriting of data in the FIFO occurs, the FIFO logic flags a FIFO overrun as the 13th byte is written into the FIFO, effectively shortening the FIFO to 13 bytes. The FIFO logic also operates differently in Byte Mode and in Word Mode. In Byte Mode, a threshold is indicated when the  $n + 1$  byte has entered the FIFO; thus, with an 8-byte threshold, the ST-NIC issues Bus Request (BREQ) when the 9th byte has entered the FIFO. For Word Mode, BREQ is not generated until the  $n + 2$  bytes have entered the FIFO. Thus, with a 4-word threshold (equivalent to 8-byte threshold), BREQ is issued when the 10th byte has entered the FIFO. The two graphs, following, indicate the maximum allowable bus latency for Word and Byte transfer modes.



TL/F/11157-58

### 13.0 Bus Arbitration and Timing (Continued)

#### The FIFO at the Beginning of Transmit

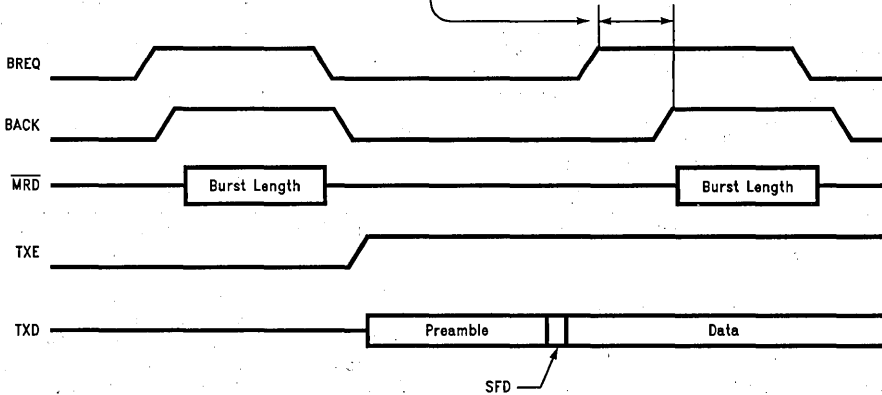
Before transmitting, the ST-NIC performs a prefetch from memory to load the FIFO. The number of bytes prefetched

is the programmed FIFO threshold. The next BREQ is not issued until after the ST-NIC actually begins transmitting data, i.e., after SFD. The Transmit Prefetch diagram illustrates this process.

**Transmit Prefetch Timing**

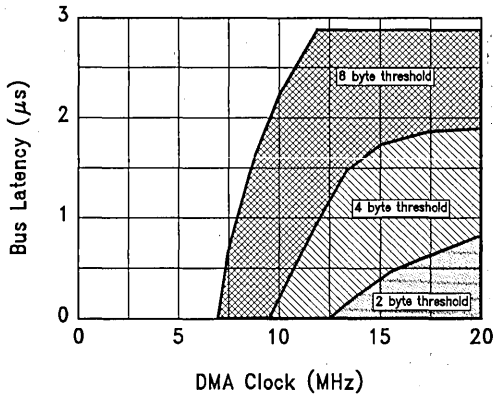
$$\text{Tolerated Bus Latency} = [(\text{No. of Bytes Stored in FIFO}) \times 800] - 400 \text{ ns}$$

$$\text{or } (12 \text{ Bytes} - \text{FIFO Threshold}) \text{ whichever is less}$$



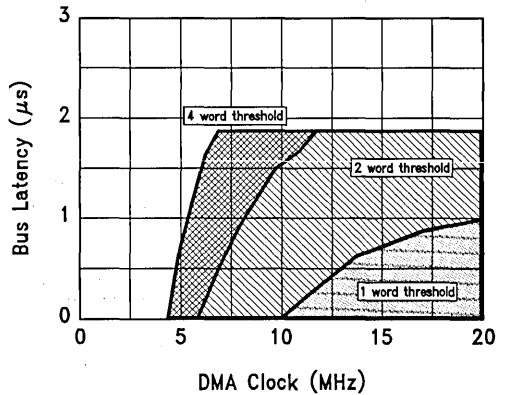
TL/F/11157-59

**Maximum Bus Latency for Byte Mode**



TL/F/11157-60

**Maximum Bus Latency for Word Mode**



TL/F/11157-61



### 13.0 Bus Arbitration and Timing (Continued)

#### REMOTE DMA-BIDIRECTIONAL PORT CONTROL

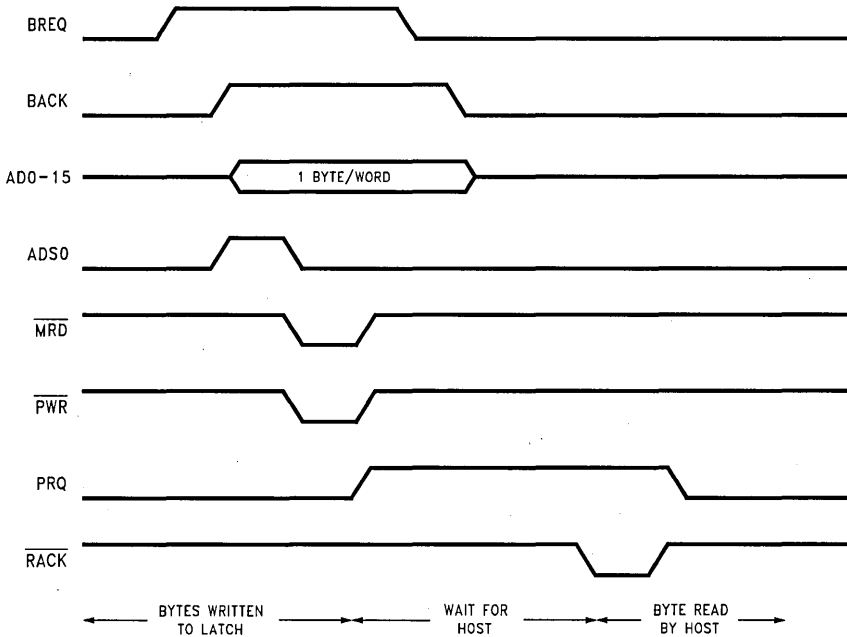
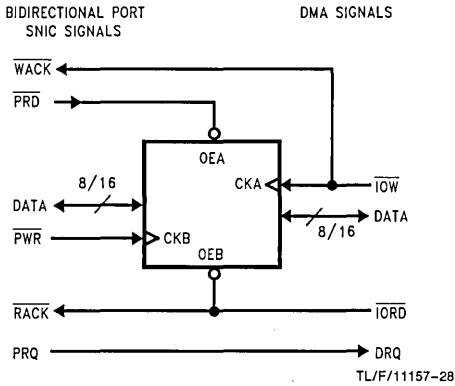
The Remote DMA transfers data between the local buffer memory and a bidirectional port (memory to I/O transfer). This transfer is arbitrated on a byte by byte basis versus the burst transfer used for Local DMA transfers. This bidirectional port is also read/written by the host. All transfers through this port are asynchronous. At any one time transfers are limited to one direction, either from the port to local buffer memory (Remote Write) or from local buffer memory to the port (Remote Read).

#### REMOTE READ TIMING

1. The DMA reads byte/word from local buffer memory and writes byte/word into latch, increments the DMA address and decrements the byte count (RBCR0, 1).
2. A Request Line (PRQ) is asserted to inform the system that a byte is available.
3. The system reads the port, the read strobe ( $\overline{RACK}$ ) is used as an acknowledge by the Remote DMA and it goes back to step 1.

Steps 1–3 are repeated until the remote DMA is complete. Note that in order for the Remote DMA to transfer a byte from memory to the latch, it must arbitrate access to the bus via a BREQ, BACK handshake. After each byte or word is transferred to the latch, BREQ is dropped. If a Local DMA is in progress, the Remote DMA is held off until the local DMA is complete.

#### Bus Handshake Signals for Remote DMA Transfers

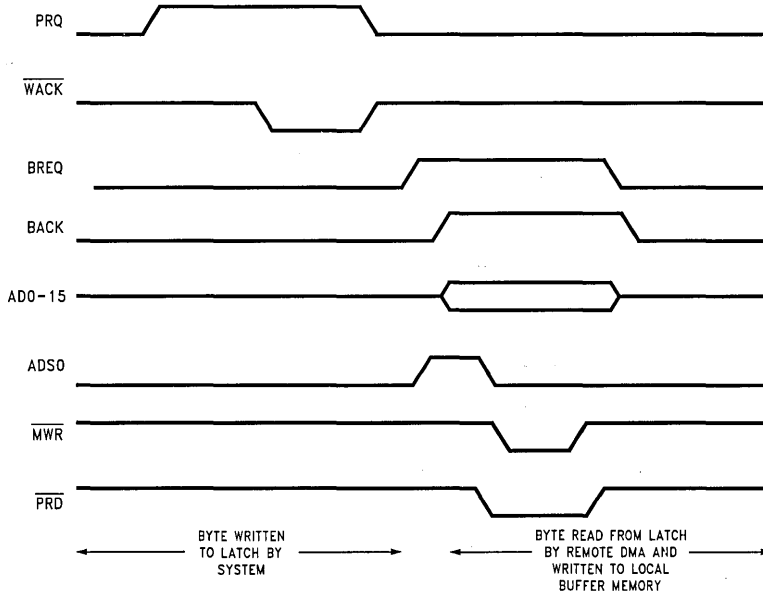


## 13.0 Bus Arbitration and Timing (Continued)

### REMOTE WRITE TIMING

A Remote Write operation transfers data from the I/O port to the local buffer RAM. The ST-NIC initiates a transfer by requesting a byte/word via the PRQ. The system transfers a byte-word to the latch via  $\overline{OW}$ . This write strobe is detected by the ST-NIC and PRQ is removed. By removing the PRQ, the Remote DMA holds off further transfers into the latch until the current byte/word has been transferred from the latch, PRQ is reasserted and the next transfer can begin.

1. ST-NIC asserts PRQ. System writes byte/word into latch. ST-NIC removes PRQ.
  2. Remote DMA reads contents of port and writes byte/word to local buffer memory, increments address and decrements byte count (RBCR0, 1).
  3. Go back to step 1.
- Steps 1-3 are repeated until the remote DMA is complete.



TL/F/11157-30

### REMOTE DMA WRITE SPECIAL CONSIDERATIONS

#### Setting PRQ Using the Remote Read

Under certain conditions the ST-NIC bus state machine may issue  $\overline{MWR}$  and  $\overline{PRD}$  before PRQ for the first DMA transfer of a Remote Write Command. If this occurs this could cause data corruption, or cause the remote DMA count to be different from the main CPU count causing the system to "lock up".

To prevent this condition when implementing a Remote DMA Write, the Remote DMA Write command should first be preceded by a Remote DMA Read command to insure that the PRQ signal is asserted before the ST-NIC starts its port read cycle. The reason for this is that the state machine that asserts PRQ runs independently of the state machine that controls the DMA signals. The DMA machine assumes that PRQ is asserted, but actually may not be. To remedy this situation, a single Remote Read cycle should be inserted before the actual DMA Write Command is given. This will ensure that PRQ is asserted when the Remote DMA Write is subsequently executed. This single Remote Read cycle is called a "dummy Remote Read". In order for the dummy Remote Read cycle to operate correctly, the Start Address should be programmed to a known, safe location in the buffer memory space, and the Remote Byte count should be programmed to a value greater than 1. This will ensure that

the master read cycle is performed safely, eliminating the possibility of data corruption.

#### Remote Write with High Speed Buses

When implementing the Remote DMA Write solution with high speed buses and CPU's, timing may cause the system to hang. Therefore additional considerations are required.

A problem occurs when the system can execute the dummy Remote Read and then start the Remote Write before the ST-NIC has had a chance to execute the Remote Read. If this happens the PRQ signal will not get set, and the Remote Byte Count and Remote Start Address for the Remote Write operation could be corrupted. This is shown by the hatched waveforms in the following timing diagram. The execution of the Remote Read can be delayed by the local DMA operations (particularly during end-of-packet processing).

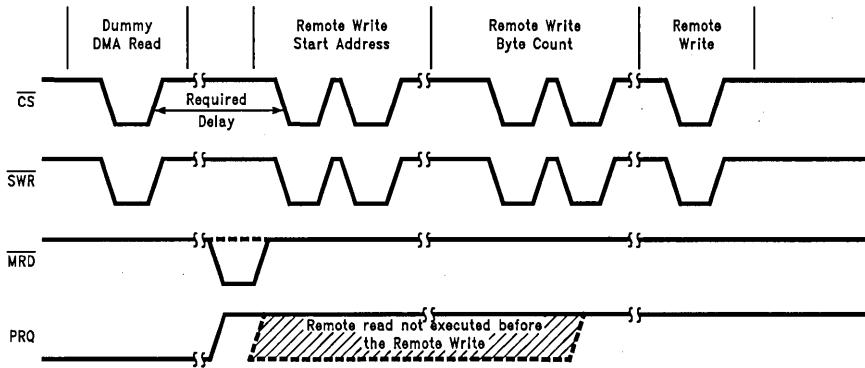
To ensure the dummy Remote Read does execute, a delay must be inserted between writing the Remote Read Command, and starting to write the Remote Write Start Address. (This time is designated in the next figure by the delay arrows.) The recommended method to avoid this problem is after the Remote Read command is given, to poll both bytes of the Current Remote DMA Address Registers. When the address has incremented PRQ has been set. Software should recognize this and then start the Remote Write.

### 13.0 Bus Arbitration and Timing (Continued)

An additional caution for high speed systems is that the polling must follow guidelines specified in the Time Between Chip Selects section. That is, there must be at least 4 bus clocks between chip selects. (For example when BSCK = 20 MHz, then this time should be 200 ns).

The general flow for executing a Remote Write is:

1. Set Remote Byte Count to a value > 1 and Remote Start Address to unused RAM (one location before the transmit start address is usually a safe location).
2. Issue the "dummy" Remote Read command.
3. Read the Current Remote DMA Address (CRDA) (both bytes).
4. Compare to previous CRDA value if different go to 6.
5. Delay and jump to 3.
6. Set up for the Remote Write command, by setting the Remote Byte Count and the Remote Start Address (note that if the Remote Byte count in step 1 can be set to the transmit byte count plus one, and the Remote Start Address to one less, these will now be incremented to the correct values.)
7. Issue the Remote Write command.



TL/F/11157-62

Note: The dashed lines indicate incorrect timing as described in text.

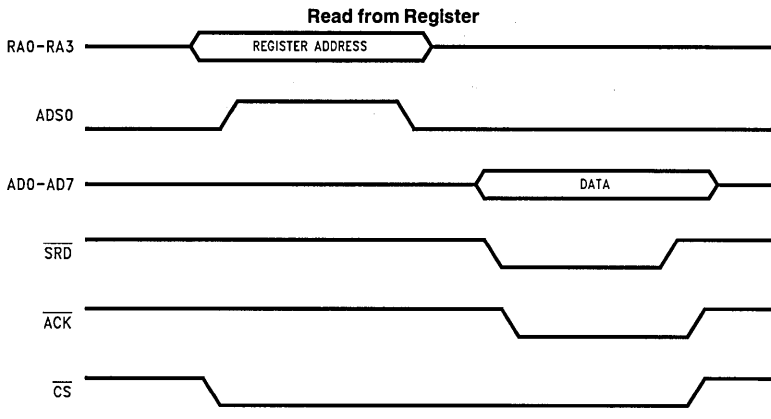
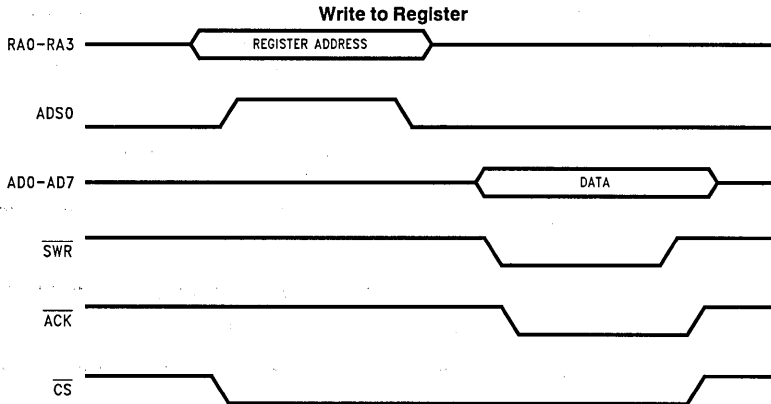
Timing Diagram for Dummy Remote Read

### 13.0 Bus Arbitration and Timing (Continued)

#### SLAVE MODE TIMING

When  $\overline{CS}$  is low, the ST-NIC becomes a bus slave. The CPU can then read or write any internal registers. All register accesses are byte wide. The timing for register access is shown below. The host CPU accesses internal registers with four address lines, RA0-RA3, SRD and SWR strobes.

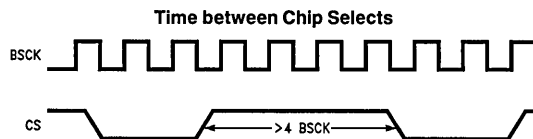
ADS0 is used to latch the address when interfacing to a multiplexed, address data bus. Since the ST-NIC may be a local bus master when the host CPU attempts to read or write to the controller, an  $\overline{ACK}$  line is used to hold off the CPU until the ST-NIC leaves master mode. Some number of BSCK cycles is also required to allow the ST-NIC to synchronize to the read or write cycles.



#### TIME BETWEEN CHIP SELECTS

The ST-NIC requires that successive chip selects be no closer than 4 bus clocks (BSCK) together. If the condition is violated, the ST-NIC may glitch  $\overline{ACK}$ . CPUs that operate from pipelined instructions (i.e., 386) or have a cache (i.e.,

486) can execute consecutive I/O cycles very quickly. The solution is to delay the execution of consecutive I/O cycles by either breaking the pipeline or forcing the CPU to access outside its cache.



## 14.0 Preliminary Electrical Characteristics

### Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage ( $V_{CC}$ )	-0.5V to +7.0V
DC Input Voltage ( $V_{IN}$ )	-0.5V to $V_{CC} + 0.5V$
DC Output Voltage ( $V_{OUT}$ )	-0.5V to $V_{CC} + 0.5V$
Storage Temperature Range ( $T_{STG}$ )	-65°C to +150°C
Power Dissipation (PD)	800 mW
Lead Temp. (TL) (Soldering, 10 sec.)	260°C
ESD Rating ( $R_{ZAP} = 1.5k$ , $C_{ZAP} = 100$ pF)	1.5 kV
Pin to Pin	
Pin to GND	
Pin to $V_{CC}$ ( $\pm 1$ ZAP)	
Clamp Diode Current	$\pm 20$ mA

Note: Absolute Maximum ratings are those values beyond which the safety of the device cannot be guaranteed. They are not meant to imply that the device should be operated at these limits.

Note: All specifications in this datasheet are valid only if the mandatory isolation is employed and all differential signals are taken to exist at the AUI or TPI side of the isolation.

### Preliminary DC Specifications $T_A = 0^\circ\text{C}$ to $70^\circ\text{C}$ , $V_{CC} = 5V \pm 5\%$ , unless otherwise specified.

Symbol	Parameter	Conditions	Min	Max	Units
$V_{OH}$	Minimum High Level Output Voltage (Notes 1, 4)	$I_{OH} = -20 \mu\text{A}$	$V_{CC} - 0.1$		V
		$I_{OH} = -2.0 \text{ mA}$			
$V_{OL}$	Minimum Low Level Output Voltage (Notes 1, 4)	$I_{OL} = 20 \mu\text{A}$		0.1	V
		$I_{OL} = 2.0 \text{ mA}$		0.4	V
$V_{IH}$	Minimum High Level Input Voltage (Note 2)		2.0		V
$V_{IH2}$	Minimum High Level Input Voltage For RACK WACK (Note 2)		2.7		V
$V_{IL}$	Maximum Low Level Input Voltage (Note 2)			0.8	V
$V_{IL2}$	Maximum Low Level Input Voltage For RACK, WACK (Note 2)			0.6	V
$V_{LOL}$	Good Link Output Voltage	$I_{OL} = 16 \text{ mA}$		0.4	V
$I_{IN}$	Input Current	$V_I = V_{CC}$ or GND	-1.0	+1.0	$\mu\text{A}$
$I_{INSEL}$	Input Current	$V_{IN} = V_{CC}$	50	2000	$\mu\text{A}$
		$V_{IN} = \text{GND}$	-1	+1	$\mu\text{A}$
$I_{OZ}$	Minimum TRI-STATE Output Leakage Current (Note 5)	$V_{OUT} = V_{CC}$ or GND	-10	+10	$\mu\text{A}$
$I_{CC}$	Average Supply Current (Note 3)	X1 = 20 MHz Clock $I_{OUT} = 0 \mu\text{A}$ $V_{IN} = V_{CC}$ or GND		140	mA

**Note 1:** These levels are tested dynamically using a limited amount of functional test patterns, please refer to AC test load.

**Note 2:** Limited functional test patterns are performed at these input levels. The majority of functional tests are performed at levels of 0V and 3V.

**Note 3:** This is measured with a 0.1  $\mu\text{F}$  bypass capacitor between  $V_{CC}$  and GND.

**Note 4:** The low drive CMOS compatible  $V_{OH}$  and  $V_{OL}$  limits are not tested directly. Detailed device characterization validates that this specification can be guaranteed by testing the high drive TTL compatible  $V_{OL}$  and  $V_{OH}$  specification.

**Note 5:** RA0-RA3, PRD, WACK, BREQ and INT pins are used as outputs in test mode and as a result are tested as if they are TRI-STATE input/outputs. For these pins the input leakage specification is  $I_{OZ}$ .

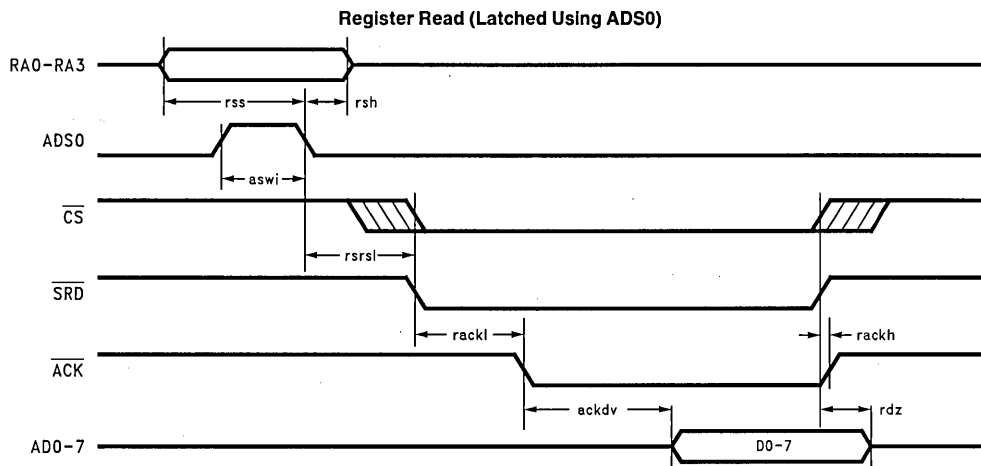
## 14.0 Preliminary Electrical Characteristics (Continued)

**Preliminary DC Specifications**  $T_A = 0^\circ\text{C to }70^\circ\text{C}$ ,  $V_{CC} = 5\text{V} \pm 5\%$ , unless otherwise specified. (Continued)

Symbol	Parameter	Conditions	Min	Max	Units
<b>AUI INTERFACE PINS (TX<math>\pm</math>, RX<math>\pm</math>, and CD<math>\pm</math>)</b>					
V <sub>OD</sub>	Diff. Output Voltage (TX $\pm$ )	78 $\Omega$ Termination, and 270 $\Omega$ from Each to GND	$\pm 550$	$\pm 1200$	mV
V <sub>OB</sub>	Diff. Output Voltage Imbalance (TX $\pm$ ) (Note 1)	78 $\Omega$ Termination, and 270 $\Omega$ from Each to GND	Typical: 40 mV		
V <sub>U</sub>	Undershoot Voltage (TX $\pm$ ) (Note 1)	78 $\Omega$ Termination, and 270 $\Omega$ from Each to GND	Typical: 80 mV		
V <sub>DS</sub>	Diff. Squelch Threshold (RX $\pm$ and CD $\pm$ ) (Note 1)		-175	-300	mV
V <sub>CM</sub>	Diff. Input Common Mode Voltage (RX $\pm$ and CD $\pm$ ) (Note 1)		0	5.25	V
<b>OSCILLATOR PINS (X1 AND X2)</b>					
V <sub>IH</sub>	X1 Input High Voltage	X1 is Connected to an Oscillator and GND/X2 is Grounded	2.0		V
V <sub>IL</sub>	X1 Input Low Voltage	X1 is Connected to an Oscillator and GND/X2 is Grounded		0.8	V
I <sub>OSC</sub>	X1 Input Current	GND/X2 is Grounded V <sub>IN</sub> = V <sub>CC</sub> or GND		3	mA
I <sub>X2</sub>	X2 Input Current	X2 Grounded (Driven Mode)		4	mA
<b>TWISTED PAIR INTERFACE PINS (TXO<math>\pm</math>, TXOd<math>\pm</math>, and RXI<math>\pm</math>)</b>					
R <sub>TOL</sub>	TXOd $\pm$ , TXO $\pm$ Low Level Output Resistance	I <sub>OL</sub> = 25 mA		15	$\Omega$
R <sub>TOH</sub>	TXOd $\pm$ , TXO $\pm$ High Level Output Resistance	I <sub>OH</sub> = 25 mA		15	$\Omega$
V <sub>SRON1</sub>	Receive Threshold Turn-On Voltage (10BASE-T)		$\pm 300$	$\pm 585$	mV
V <sub>SRON2</sub>	Receive Threshold Turn-On Voltage (Reduced Level)		$\pm 175$	$\pm 300$	mV
V <sub>SROFF</sub>	Receive Threshold Turn-Off Voltage (Note 1)		$\pm 175$	V <sub>SRON</sub> - 100	mV
V <sub>DIFF</sub>	Differential Mode Input Voltage Range (Note 1)	V <sub>CC</sub> = 5.0V	-3.1	+3.1	V

**Note 1:** This parameter is guaranteed by design and is not tested.

### 15.0 Switching Characteristics AC Specs DP83902A Note: All Timing is Preliminary



TL/F/11157-33

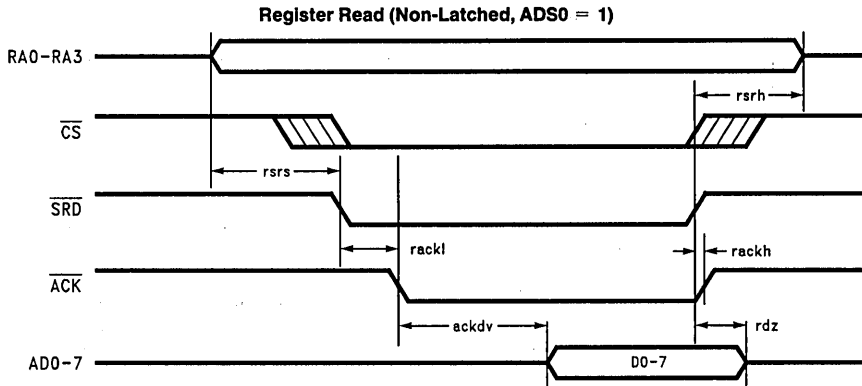
Symbol	Parameter	Min	Max	Units
rss	Register Select Setup to ADS0 Low	10		ns
rsh	Register Select Hold from ADS0 Low	13		ns
aswi	Address Strobe Width In	15		ns
ackdv	Acknowledge Low to Data Valid		55	ns
rdz	Read Strobe to Data TRI-STATE (Note 3)	15	70	ns
rackl	Read Strobe to $\overline{\text{ACK}}$ Low (Notes 1, 2)		$n \cdot \text{bcyc} + 30$	ns
rackh	Read Strobe to $\overline{\text{ACK}}$ High		30	ns
rsrsl	Register Select to Slave Read Low, Latched RS0-3	10		ns

**Note 1:**  $\overline{\text{ACK}}$  is not generated until  $\overline{\text{CS}}$  and  $\overline{\text{SRD}}$  are low and the ST-NIC has synchronized to the register access. The ST-NIC will insert an integral number of Bus Clock cycles until it is synchronized. In Dual Bus systems additional cycles will be used for a local or remote DMA to complete. Wait states must be issued to the CPU until  $\overline{\text{ACK}}$  is asserted low.

**Note 2:**  $\overline{\text{CS}}$  may be asserted before or after  $\overline{\text{SRD}}$ . If  $\overline{\text{CS}}$  is asserted after  $\overline{\text{SRD}}$ , rackl is referenced from falling edge of  $\overline{\text{CS}}$ .  $\overline{\text{CS}}$  can be de-asserted concurrently with  $\overline{\text{SRD}}$  or after  $\overline{\text{SRD}}$  is de-asserted.

**Note 3:** These limits include the RC delay inherent in our test method. These signals typically turn off within 15 ns, enabling other devices to drive these lines with no contention.

**15.0 Switching Characteristics** AC Specs DP83902A Note: All Timing is Preliminary (Continued)



TL/F/11157-34

Symbol	Parameter	Min	Max	Units
rsrc	Register Select to Read Setup (Notes 1, 3)	10		ns
rsrh	Register Select Hold from Read	0		ns
ackdv	ACK Low to Valid Data		55	ns
rdz	Read Strobe to Data TRI-STATE (Note 2)	15	70	ns
rackl	Read Strobe to ACK Low (Note 3)		$n \cdot bcyc + 30$	ns
rackh	Read Strobe to ACK High		30	ns

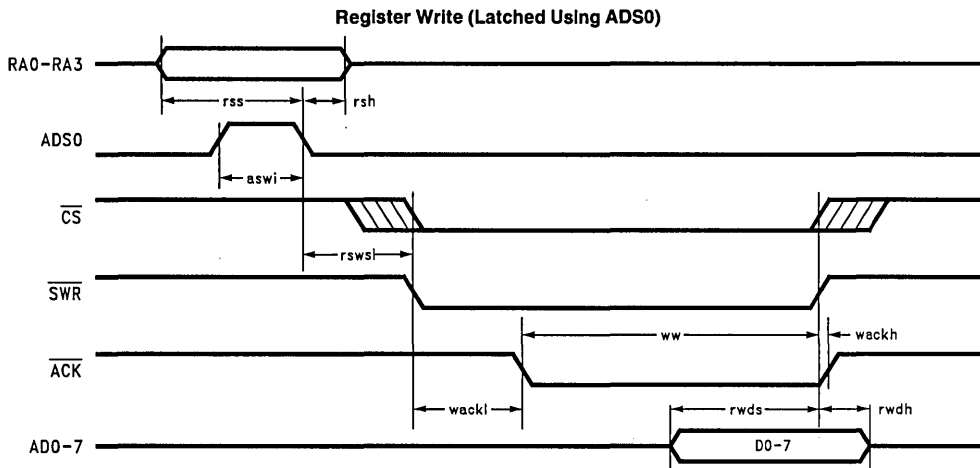
**Note 1:** rsrc includes flow-through time of latch.

**Note 2:** These limits include the RC delay inherent in our test method. These signals typically turn off within 15 ns enabling other devices to drive these lines with no contention.

**Note 3:** CS may be asserted before or after RA0-3, and SRD, since address decode begins when ACK is asserted. If CS is asserted after RA0-3, and SRD, rackl is referenced from falling edge of CS.



**15.0 Switching Characteristics** AC Specs DP83902A **Note:** All Timing is Preliminary (Continued)



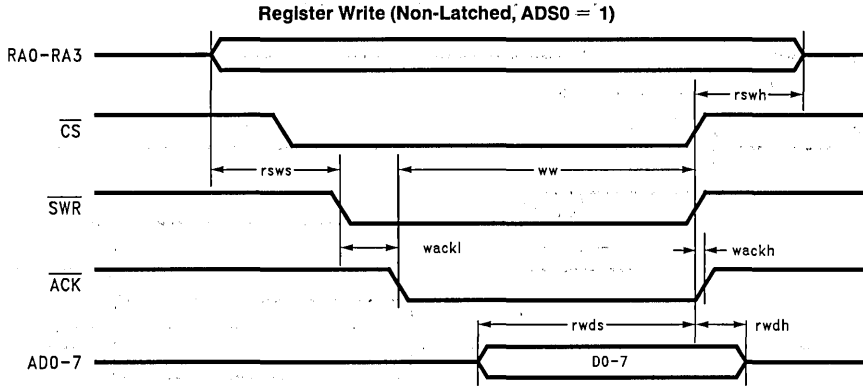
TL/F/11157-35

Symbol	Parameter	Min	Max	Units
rss	Register Select Setup to ADS0 Low	10		ns
rsh	Register Select Hold from ADS0 Low	17		ns
aswi	Address Strobe Width In	15		ns
rwds	Register Write Data Setup	20		ns
rwdh	Register Write Data Hold	21		ns
ww	Write Strobe Width from $\overline{ACK}$	50		ns
wackh	Write Strobe High to $\overline{ACK}$ High		30	ns
wackl	Write Low to $\overline{ACK}$ Low (Notes 1, 2)		$n \cdot \text{bcyc} + 30$	ns
rswsl	Register Select to Write Strobe Low	10		ns

**Note 1:**  $\overline{ACK}$  is not generated until  $\overline{CS}$  and  $\overline{SWR}$  are low and the ST-NIC has synchronized to the register access. In Dual Bus Systems additional cycles will be used for a local DMA or Remote DMA to complete.

**Note 2:**  $\overline{CS}$  may be asserted before or after  $\overline{SWR}$ . If  $\overline{CS}$  is asserted after  $\overline{SWR}$ , wackl is referenced from falling edge of  $\overline{CS}$ .

**15.0 Switching Characteristics** AC Specs DP83902A Note: All Timing is Preliminary (Continued)



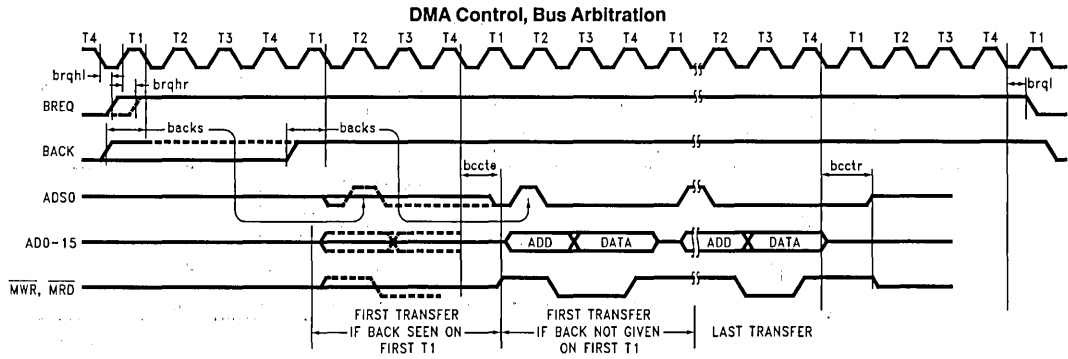
TL/F/11157-36

Symbol	Parameter	Min	Max	Units
rsws	Register Select to Write Setup (Note 1)	15		ns
rswh	Register Select Hold from Write	0		ns
rws	Register Write Data Setup	20		ns
rwdh	Register Write Data Hold	21		ns
wackl	Write Low to ACK Low (Note 2)		$n \cdot \text{bcyc} + 30$	ns
wackh	Write High to ACK High		30	ns
ww	Write Width from ACK	50		ns

**Note 1:** Assumes ADS0 is high when RA0-3 changing.

**Note 2:** ACK is not generated until CS and SWR are low and the ST-NIC has synchronized to the register access. In Dual Bus systems additional cycles will be used for a local DMA or remote DMA to complete.

**15.0 Switching Characteristics** AC Specs DP83902A **Note:** All Timing is Preliminary (Continued)



TL/F/11157-37

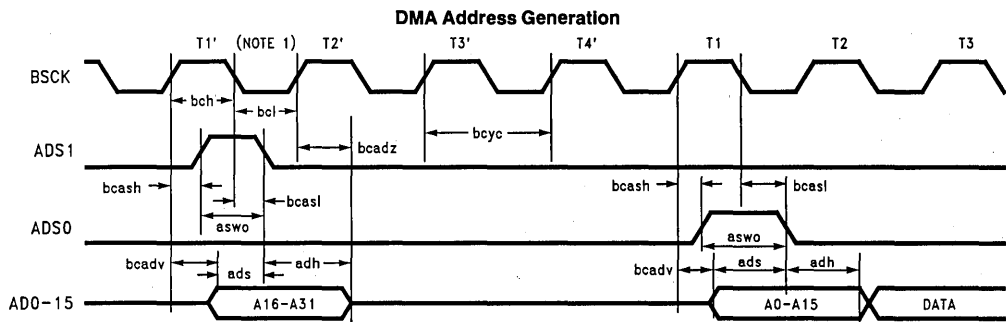
Symbol	Parameter	Min	Max	Units
brqhl	Bus Clock to Bus Request High for Local DMA		50	ns
brqhr	Bus Clock to Bus Request High for Remote DMA		45	ns
brql	Bus Request Low from Bus Clock		60	ns
backs	Acknowledge Setup to Bus Clock (Note 1)	2		ns
bccte	Bus Clock to Control Enable		60	ns
bcctr	Bus Clock to Control Release (Notes 2, 3)		70	ns

**Note 1:** BACK must be setup before T1 after BREQ is asserted. Missed setup will slip the beginning of the DMA by four bus clocks. The Bus Latency will influence the allowable FIFO threshold.

**Note 2:** During remote DMA transfers only, a single bus transfer is performed. During local DMA operations burst mode transfers are performed.

**Note 3:** These limits include the RC delay inherent in our test method. These signals typically turn off within 15 ns enabling other devices to drive these lines with no contention.

# 15.0 Switching Characteristics AC Specs DP83902A Note: All Timing is Preliminary (Continued)



TL/F/11157-38

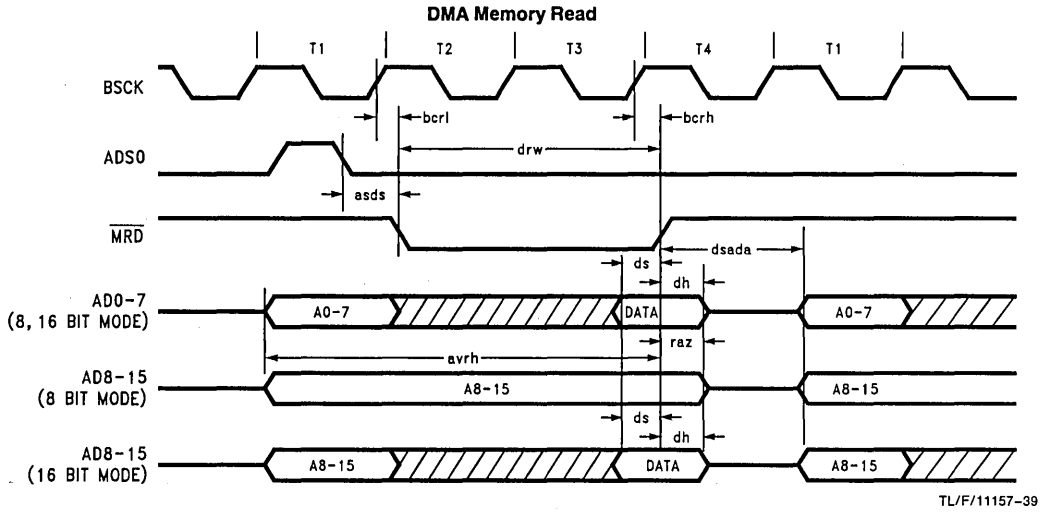
Symbol	Parameter	Min	Max	Units
bcyc	Bus Clock Cycle Time (Note 2)	50	125	ns
bch	Bus Clock High Time	20		ns
bcl	Bus Clock Low Time	20		ns
bcash	Bus Clock to Address Strobe High		34	ns
bcasl	Bus Clock to Address Strobe Low		44	ns
aswo	Address Strobe Width Out	bch		ns
bcadv	Bus Clock to Address Valid		45	ns
bcadz	Bus Clock to Address TRI-STATE (Note 3)	15	55	ns
ads	Address Setup to ADS0/1 Low	bch - 15		ns
adh	Address Hold from ADS0/1 Low	bcl - 5		ns

**Note 1:** Cycles T1', T2', T3' and T4' are only issued for the first transfer in a burst when 32-bit mode has been selected.

**Note 2:** The rate of bus clock must be high enough to support transfers to/from the FIFO at a rate greater than the serial network transfers from/to the FIFO.

**Note 3:** These limits include the RC delay inherent in our test method. These signals typically turn off within 15 ns, enabling other devices to drive these lines with no contention.

**15.0 Switching Characteristics** AC Specs DP83902A **Note:** All Timing is Preliminary (Continued)

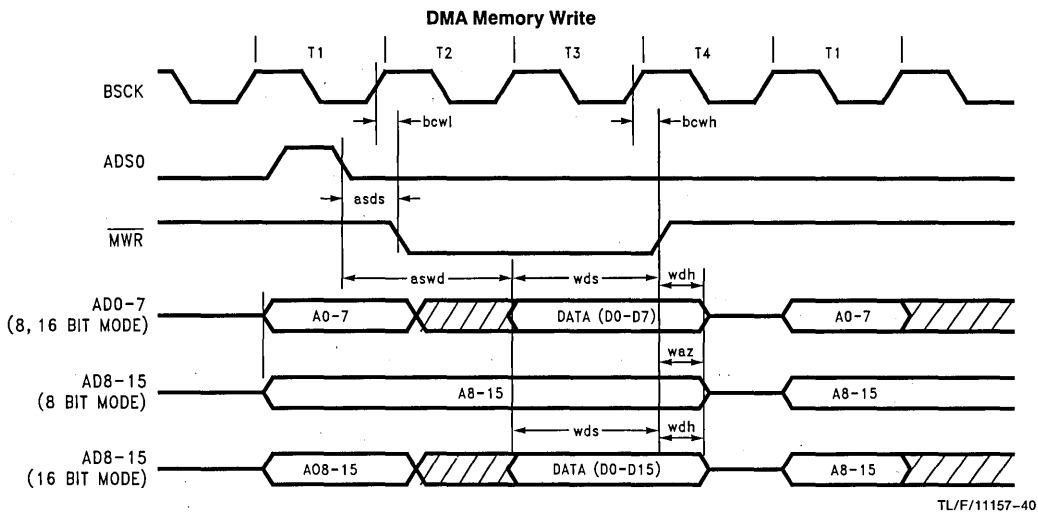


Symbol	Parameter	Min	Max	Units
bcr1	Bus Clock to Read Strobe Low		43	ns
bcrh	Bus Clock to Read Strobe High		40	ns
ds	Data Setup to Read Strobe High	22		ns
dh	Data Hold from Read Strobe High	0		ns
drw	DMA Read Strobe Width Out	$2 \cdot \text{bcyc} - 15$		ns
raz	Memory Read High to Address TRI-STATE (Notes 1, 2)		$\text{bch} + 40$	ns
asds	Address Strobe to Data Strobe		$\text{bcl} + 10$	ns
dsada	Data Strobe to Address Active	$\text{bcyc} - 10$		ns
avrh	Address Valid to Read Strobe High	$3 \cdot \text{bcyc} - 18$		ns

**Note 1:** During a burst A8-A15 are not TRI-STATE if byte wide transfers are selected. On the last transfer A8-A15 are TRI-STATE as shown above.

**Note 2:** These limits include the RC delay inherent in our test method. These signals typically turn off within  $\text{bch} + 15$  ns, enabling other devices to drive these lines with no contention.

# 15.0 Switching Characteristics AC Specs DP83902A Note: All Timing is Preliminary (Continued)

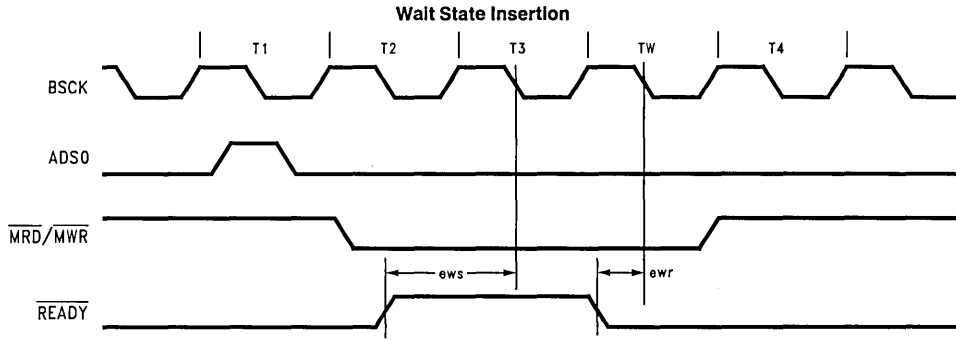


TL/F/11157-40

Symbol	Parameter	Min	Max	Units
bcwl	Bus Clock to Write Strobe Low		40	ns
bcwh	Bus Clock to Write Strobe High		40	ns
wds	Data Setup to MWR High	$2 * bcyc - 30$		ns
wdh	Data Hold from MWR Low	$bch + 7$		ns
waz	Write Strobe to Address TRI-STATE (Notes 1, 2)		$bch + 40$	ns
asds	Address Strobe to Data Strobe		$bcl + 10$	ns
aswd	Address Strobe to Write Data Valid		$bcl + 30$	ns

**Note 1:** When using byte mode transfers A8-A15 are only TRI-STATE on the last transfer, waz timing is only valid for last transfer in a burst.  
**Note 2:** These limits include the RC delay inherent in our test method. These signals typically turn off within  $bch + 15$  ns, enabling other devices to drive these lines with no contention.

# 15.0 Switching Characteristics AC Specs DP83902A Note: All Timing is Preliminary (Continued)



TL/F/11157-41

Symbol	Parameter	Min	Max	Units
ews	External Wait Setup to T3 0Clock (Note 1)	10		ns
ewr	External Wait Release Time (Note 1)	15		ns

**Note 1:** The addition of wait states affects the count of deserialized bytes and is limited to a number of bus clock cycles depending on the bus clock and network rates. The allowable wait states are found in the table below. (Assumes 10 Mbit/sec data rate.)

BSCCK (MHz)	Max # of Wait States	
	Byte Transfer	Word Transfer
8	0	1
10	0	1
12	1	2
14	1	2
16	1	3
18	2	3
20	2	4

Table assumes 10 MHz network clock.

The number of allowable wait states in byte mode can be calculated using:

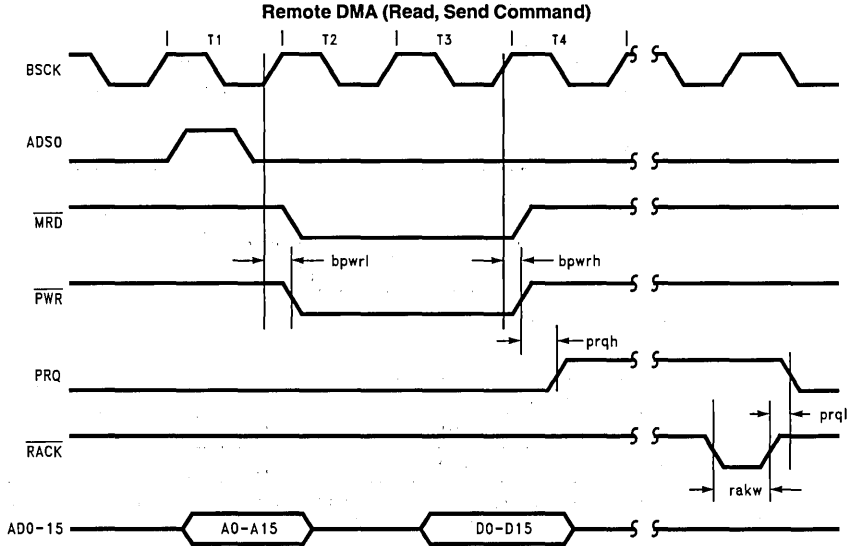
$$\#W_{(\text{byte mode})} = \left( \frac{8 \text{tnw}}{4.5 \text{tbsck}} - 1 \right)$$

- #W = Number of Wait States
- tnw = Network Clock Period
- tbsck = BSCCK Period

The number of allowable wait states in word mode can be calculated using:

$$\#W_{(\text{word mode})} = \left( \frac{5 \text{tnw}}{2 \text{tbsck}} - 1 \right)$$

# 15.0 Switching Characteristics AC Specs DP83902A Note: All Timing is Preliminary (Continued)



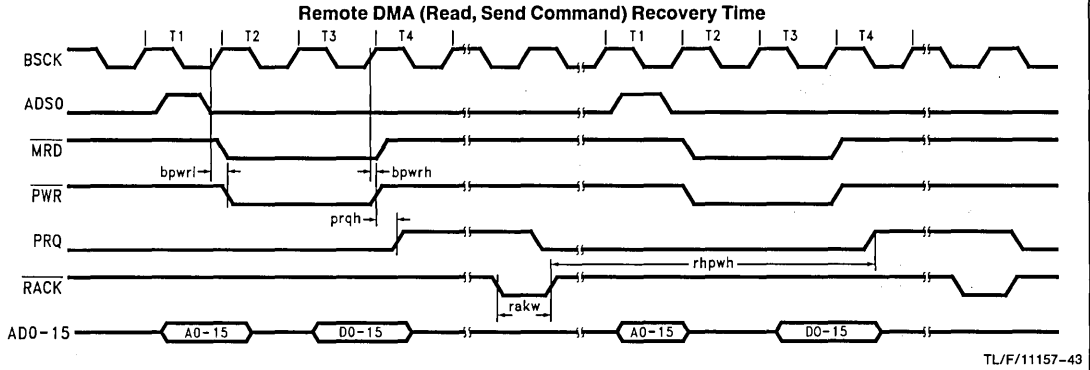
TL/F/11157-42

Symbol	Parameter	Min	Max	Units
bpwrl	Bus Clock to Port Write Low		43	ns
bpwrh	Bus Clock to Port Write High		40	ns
prqh	Port Write High to Port Request High (Note 1)		30	ns
prql	Port Request Low from Read Acknowledge High		60	ns
rakw	Remote Acknowledge Read Strobe Pulse Width	20		ns

Note 1: Start of next transfer is dependent on where RACK is generated relative to BSCCK and whether a local DMA is pending.



**15.0 Switching Characteristics** AC Specs DP83902A **Note:** All Timing is Preliminary (Continued)



Symbol	Parameter	Min	Max	Units
bpwrl	Bus Clock to Port Write Low		43	ns
bpwrh	Bus Clock to Port Write High		40	ns
prqh	Port Write High to Port Request High (Note 1)		30	ns
prql	Port Request Low from Read Acknowledge High		60	ns
rakw	Remote Acknowledge Read Strobe Pulse Width	20		ns
rhpwh	Read Acknowledge High to Next Port Write Cycle (Notes 2, 3, 4)	11		BSCK

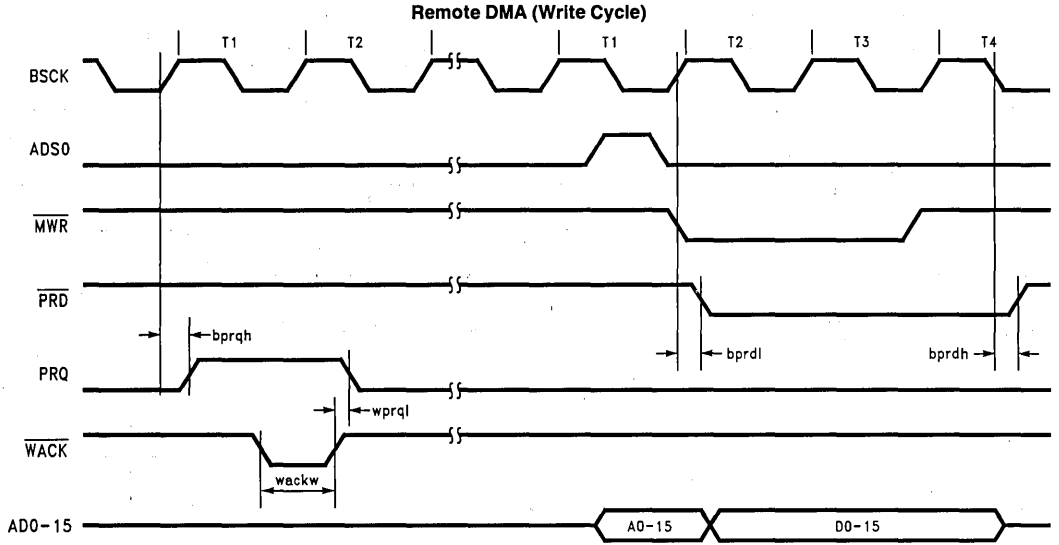
**Note 1:** Start of next transfer is dependent on where RACK is generated relative to BSCK and whether or not a local DMA is pending.

**Note 2:** This is not a measured value but guaranteed by design.

**Note 3:** RACK must be high for a minimum of 7 BSCK.

**Note 4:** Assumes no local DMA interleave, no CS, and immediate BACK.

**15.0 Switching Characteristics** AC Specs DP83902A **Note:** All Timing is Preliminary (Continued)



TL/F/11157-44

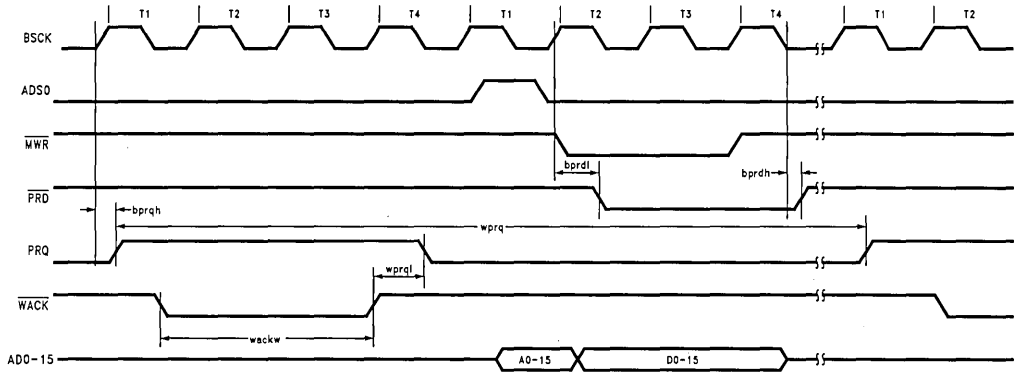
Symbol	Parameter	Min	Max	Units
bprqh	Bus Clock to Port Request High (Note 1)		42	ns
wprql	WACK to Port Request Low		52	ns
wackw	WACK Pulse Width	25		ns
bprdl	Bus Clock to Port Read Low (Note 2)		55	ns
bprdh	Bus Clock to Port Read High		40	ns

**Note 1:** The first port request is issued in response to the remote write command. It is subsequently issued on T1 clock cycles following completion of remote DMA cycles.

**Note 2:** The start of the remote DMA write following WACK is dependent on where WACK is issued relative to BSCk and whether a local DMA is pending.

### 15.0 Switching Characteristics AC Specs DP83902A Note: All Timing is Preliminary (Continued)

Remote DMA (Write Cycle) Recovery Time



TL/F/11157-45

Symbol	Parameter	Min	Max	Units
bprqh	Bus Clock to Port Request High (Note 1)		42	ns
wprql	$\overline{WACK}$ to Port Request Low		50	ns
wackw	$\overline{WACK}$ Pulse Width	25		ns
bprdl	Bus Clock to Port Read Low (Note 2)		55	ns
bprdh	Bus Clock to Port Read High		40	ns
wprq	Remote Write Port Request to Port Request Time (Notes 3, 4, 5)	12		BSCK

**Note 1:** The first port request is issued in response to the remote write command. It is subsequently issued on T1 clock cycles following completion of remote DMA cycles.

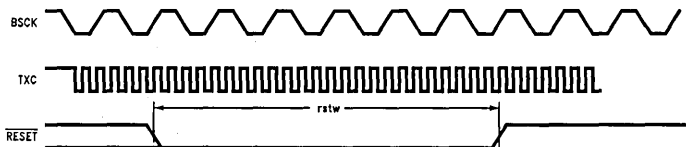
**Note 2:** The start of the remote DMA write following  $\overline{WACK}$  is dependent on where  $\overline{WACK}$  is issued relative to BSKC and whether a local DMA is pending.

**Note 3:** Assuming wackw < 1 BSCK, and no local DMA interleave, no  $\overline{CS}$ , immediate  $\overline{BACK}$ , and  $\overline{WACK}$  goes high before T4.

**Note 4:**  $\overline{WACK}$  must be high for a minimum of 7 BSCK.

**Note 5:** This is not a measured value but guaranteed by design.

Reset Timing



TL/F/11157-64

Symbol	Parameter	Min	Max	Units
rstw	Reset Pulse Width (Note 1)	8		BSCK Cycles or TXC Cycles (Note 2)

**Note 1:** The RESET pulse requires that BSKC and TXC be stable. On power up,  $\overline{RESET}$  should not be raised until BSKC and TXC have become stable. Several registers are affected by  $\overline{RESET}$ . Consult the register descriptions for details.

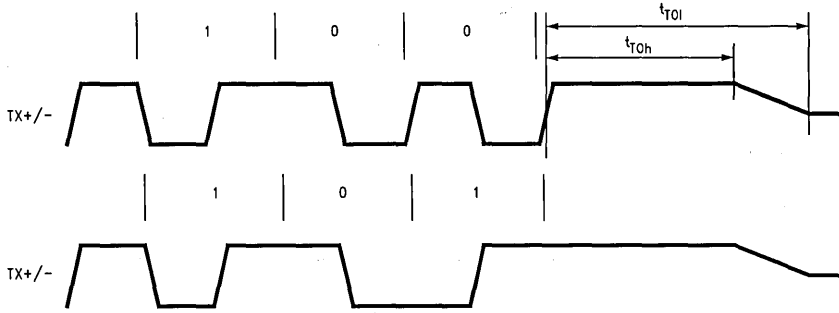
**Note 2:** The slower of BSKC or TXC clocks will determine the minimum time for the  $\overline{RESET}$  signal to be low.

If BSKC < TXC then  $\overline{RESET} = 8 \times \text{BSCK}$

If TXC < BSKC then  $\overline{RESET} = 8 \times \text{TXC}$

**15.0 Switching Characteristics** AC Specs DP83902A **Note:** All Timing is Preliminary (Continued)

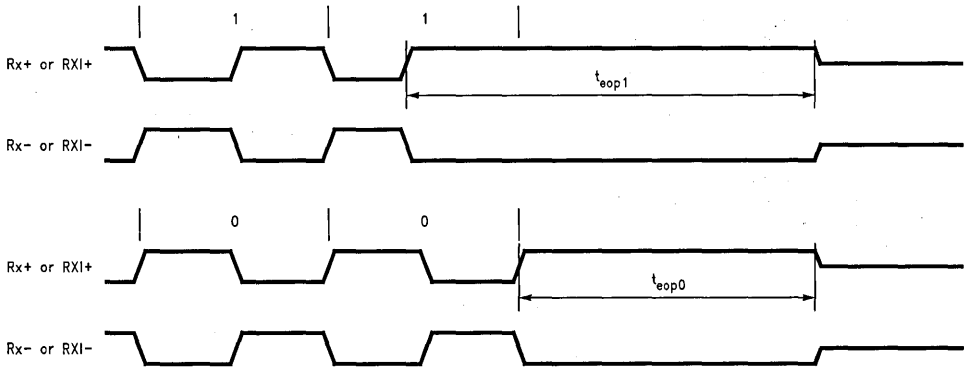
**AUI Transmit Timing (End of Packet)**



TL/F/11157-46

Symbol	Parameter	Min	Max	Units
$t_{TOh}$	Transmit Output High before Idle (Half Step)	200		ns
$t_{TOI}$	Transmit Output Idle Time (Half Step)		8000	ns

**AUI/TPI Receive End of Packet Timing**



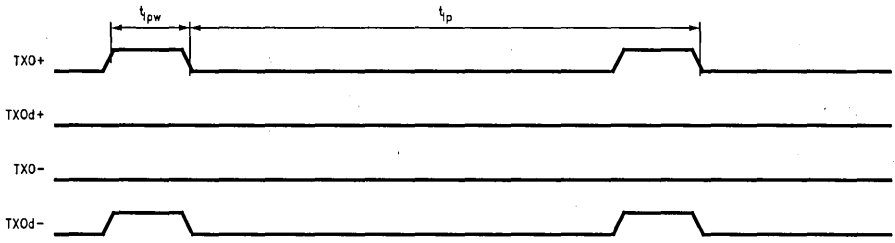
TL/F/11157-47

Symbol	Parameter	Min	Max	Units
$teop1$	Receive End of Packet Hold Time after Logic "1" (Note 1)	225		ns
$teop0$	Receive End of Packet Hold Time after Logic "0" (Note 1)	225		ns

**Note 1:** This parameter is guaranteed by design and is not tested.

**15.0 Switching Characteristics** AC Specs DP83902A Note: All Timing is Preliminary (Continued)

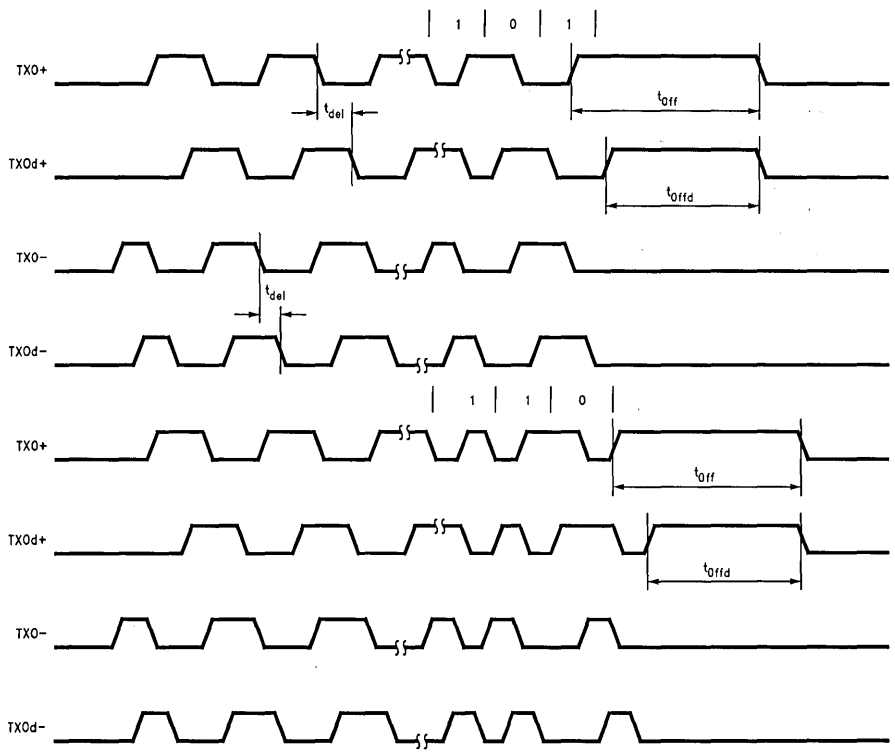
**Link Pulse Timing**



TL/F/11157-48

Symbol	Parameter	Min	Max	Units
tip	Time between Link Output Pulses	8	24	ms
tipw	Link Integrity Output Pulse Width	80	130	ns

**TPI Transmit and End of Packet Timing**



TL/F/11157-49

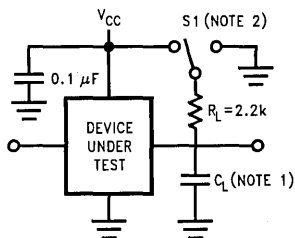
Symbol	Parameter	Min	Max	Units
tdel	Pre-Emphasis Output Delay (TX0± to TX0d±) (Note 1)	46	54	ns
toff	Transmit Hold Time at End of Packet (TX0±) (Note 1)	250		ns
toffd	Transmit Hold Time at End of Packet (TX0d±) (Note 1)	200		ns

**Note 1:** This parameter is guaranteed by design and is not tested.

## 16.0 AC Timing Test Conditions

All specifications are valid only if the mandatory isolation is employed and all differential signals are taken to be at the AUI side of the pulse transformer.

Input Pulse Levels (TTL/CMOS)	GND to 3.0V
Input Rise and Fall Times (TTL/CMOS)	5 ns
Input and Output Reference Levels (TTL/CMOS)	1.3V
Input Pulse Levels (Diff.)	-350 mV to -1315 mV
Input and Output Reference Levels (Diff.)	50% Point of the Differential
TRI-STATE Reference Levels	Float ( $\Delta V$ ) $\pm 0.5V$
Output Load (See Figure Below)	



TL/F/11157-50

**Note 1:** 50 pF, includes scope and jig capacitance.

**Note 2:** S1 = Open for timing tests for push pull outputs.

S1 = V<sub>CC</sub> for V<sub>OL</sub> test.

S1 = GND for V<sub>OH</sub> test.

S1 = V<sub>CC</sub> for High Impedance to active low and active low to High Impedance measurements.

S1 = GND for High Impedance to active high and active high to High Impedance measurements.

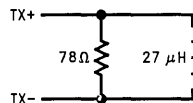
## Pin Capacitance $T_A = 25^\circ C, f = 1 \text{ MHz}$

Symbol	Parameter	Typ	Units
C <sub>IN</sub>	Input Capacitance	7	pF
C <sub>OUT</sub>	Output Capacitance	7	pF

### DERATING FACTOR

Output timing is measured with a purely capacitive load of 50 pF. The following correction factor can be used for other loads:  $C_L \geq 50 \text{ pF} + 0.3 \text{ ns/pF}$ .

### AUI Transmit Test Load



TL/F/11157-51

**Note:** In the above diagram, the TX+ and TX- signals are taken from the AUI side of the isolation (pulse transformer). The pulse transformer used for all testing is the Pulse Engineering PE64103.

# DP83901A SNIC

## Serial Network Interface Controller

### General Description

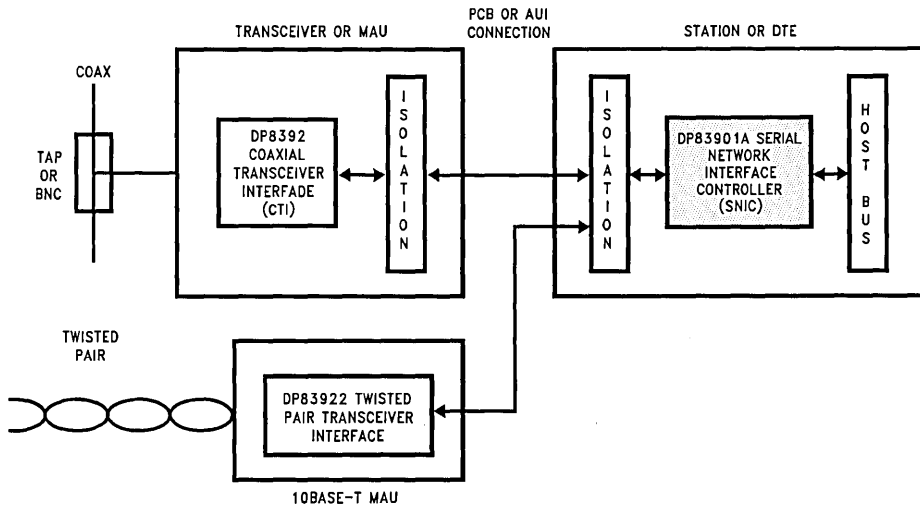
The DP83901A Serial Network Interface Controller (SNIC) is a microCMOS VLSI device designed for easy implementation of CSMA/CD local area networks. These include Ethernet (10BASE5), Thin Ethernet (10BASE2) and Twisted-pair Ethernet (10BASE-T). The overall SNIC solution provides the Media Access Control (MAC) and Encode-Decode (ENDEC) functions in accordance with the IEEE 802.3 standard.

The integrated ENDEC module allows Manchester encoding and decoding via a differential transceiver and phase lock loop at 10 Mbit/sec. Also included is a collision detect translator and diagnostic loopback capability. (Continued)

### Features

- Compatible with IEEE 802.3, 10BASE5, 10BASE2, 10BASE-T
- Dual 16-byte DMA channels
- 16-byte internal FIFO
- Network statistics storage
- Supports physical, multicast and broadcast address filtering
- 10 Mbit/sec Manchester encoding and decoding plus clock recovery
- No external precision components required
- Efficient buffer management implementation
- Transmitter can be selected for half or full step mode
- Integrated squelch on receive and collision pairs
- 3 levels of loopback supported
- Utilizes independent system and network clocks
- Lock Time 5 bits typical
- Decodes Manchester data with up to  $\pm 18$  ns jitter

### 1.0 System Diagram



TL/F/10469-1

## General Description (Continued)

The MAC function (NIC) provides simple and efficient packet transmission and reception control by means of unique dual DMA channels and an internal FIFO. Bus arbitration and memory control logic are integrated to reduce board cost and area overheads.

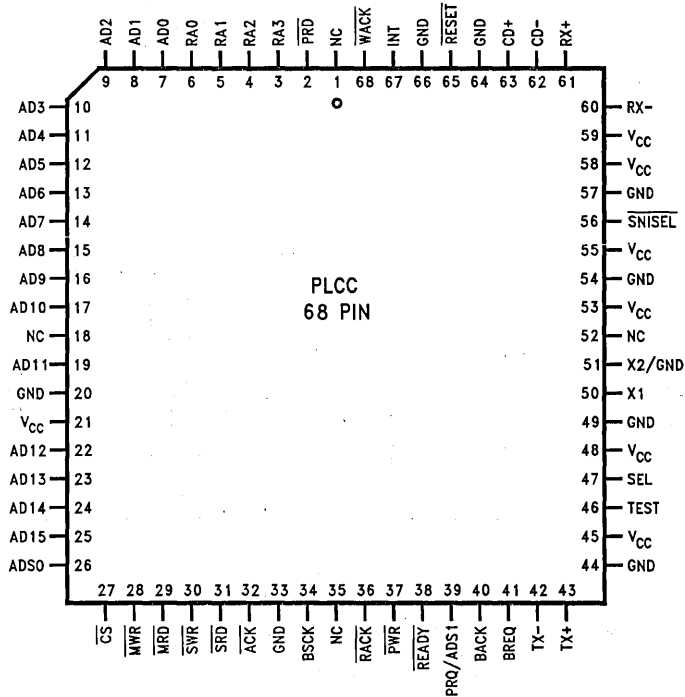
SNIC used in conjunction with the DP8392 Coaxial Transceiver Interface (CTI) provides a comprehensive 2 chip solution for IEEE 802.3 networks and is designed for easy interface to the latest 10BASE-T transceivers.

Due to the inherent constraints of CMOS processing, isolation is required at the differential signal interfaces for 10BASE5 and 10BASE2 applications. Capacitive or inductive isolation may be used.

## Table Of Contents

- 1.0 SYSTEM DIAGRAM
- 2.0 PIN DESCRIPTION
- 3.0 BLOCK DIAGRAM
- 4.0 FUNCTIONAL DESCRIPTION
- 5.0 TRANSMIT/RECEIVE PACKET ENCAPSULATION/DECAPSULATION
- 6.0 DIRECT MEMORY ACCESS CONTROL (DMA)
- 7.0 PACKET RECEPTION
- 8.0 PACKET TRANSMISSION
- 9.0 REMOTE DMA
- 10.0 INTERNAL REGISTERS
- 11.0 INITIALIZATION PROCEDURE
- 12.0 LOOPBACK DIAGNOSTICS
- 13.0 BUS ARBITRATION
- 14.0 PRELIMINARY ELECTRICAL CHARACTERISTICS
- 15.0 SWITCHING CHARACTERISTICS
- 16.0 AC TIMING TEST CONDITIONS
- 17.0 PHYSICAL DIMENSIONS

## Connection Diagram



Top View

Order Number DP83901AV  
See NS Package Number V68A

TL/F/10469-2



## Pin Description

Pin No	Pin Name	I/O	Description
<b>BUS INTERFACE PINS</b>			
2	$\overline{\text{PRD}}$	O	<b>PORT READ:</b> Enables data from external latch on to local bus during a memory write cycle to local memory (remote write operation). This allows asynchronous transfer of data from the system memory to local memory.
3–6	RA0–RA3	I	<b>REGISTER ADDRESS:</b> These four pins are used to select a register to be read or written. The state of these inputs is ignored when the NIC is not in slave mode ( $\overline{\text{CS}}$ high).
7–17, 19, 22–25	AD0–AD15	I/O, Z	<b>MULTIPLEXED ADDRESS/DATA BUS:</b> <ul style="list-style-type: none"> <li>Register Access, with DMA inactive, <math>\overline{\text{CS}}</math> low and <math>\overline{\text{ACK}}</math> returned from SNIC, pins AD0–AD7 are used to read and write register data. AD8–AD15 float during I/O transfers, <math>\overline{\text{SRD}}</math>, <math>\overline{\text{SWR}}</math> pins are used to select direction of transfer.</li> <li>Bus Master with BACK input asserted.           <ul style="list-style-type: none"> <li>During t1 of memory cycle AD0–AD15 contain address.</li> <li>During t2, t3, t4 AD0–AD15 contain data (word transfer mode).</li> <li>During t2, t3, t4 AD0–AD7 contain data, AD8–AD15 contain address (byte transfer mode).</li> </ul> </li> </ul> Direction of transfer is indicated by SNIC on $\overline{\text{MWR}}$ , $\overline{\text{MRD}}$ lines.
26	ADS0	I/O, Z	<b>ADDRESS STROBE 0:</b> <ul style="list-style-type: none"> <li>Input: with DMA inactive and <math>\overline{\text{CS}}</math> low, latches RA0–RA3 inputs on falling edge. If high, data present on RA0–RA3 will flow through latch.</li> <li>Output: When Bus Master, latches address bits (A0–A15) to external memory during DMA transfers.</li> </ul>
27	$\overline{\text{CS}}$	O	<b>CHIP SELECT:</b> Chip Select places controller in slave mode for $\mu\text{P}$ access to internal registers. Must be valid through data portion of bus cycle. RA0–RA3 are used to select the internal register. $\overline{\text{SWR}}$ and $\overline{\text{SRD}}$ select direction of data transfer.
28	$\overline{\text{MWR}}$	O, Z	<b>MASTER WRITE STROBE:</b> (Strobe for DMA transfers) Active low during write cycles (t2, t3, tw) to buffer memory. Rising edge coincides with the presence of valid output data. TRI-STATE® until BACK asserted.
29	$\overline{\text{MRD}}$	O, Z	<b>MASTER READ STROBE:</b> (Strobe for DMA transfers) Active during read cycles (t2, t3, tw) to buffer memory. Input data must be valid on rising edge of $\overline{\text{MRD}}$ . TRI-STATE until BACK asserted.
30	$\overline{\text{SWR}}$	I	<b>SLAVE WRITE STROBE:</b> Strobe from CPU to write an internal register selected by RA0–RA3. Data is latched into the SNIC on the rising edge of this input.
31	$\overline{\text{SRD}}$	I	<b>SLAVE READ STROBE:</b> Strobe from CPU to read an internal register selected by RA0–RA3. The register data is output when $\overline{\text{SRD}}$ goes low.
32	$\overline{\text{ACK}}$	O	<b>ACKNOWLEDGE:</b> Active low when SNIC grants access to CPU. Used to insert WAIT states to CPU until SNIC is synchronized for a register read or write operation.
34	BCK	I	<b>BUS CLOCK:</b> This clock is used to establish the period of the DMA memory cycle. Four clock cycles (t1, t2, t3, t4) are used per DMA cycle. DMA transfers can be extended by one BCK increments using the READY input.
36	$\overline{\text{RACK}}$	I	<b>READ ACKNOWLEDGE:</b> Indicates that the system DMA or host CPU has read the data placed in the external latch by the SNIC. The SNIC will begin a read cycle to update the latch.
37	$\overline{\text{PWR}}$	O	<b>PORT WRITE:</b> Strobe used to latch data from the SNIC into external latch for transfer to host memory during Remote Read transfers. The rising edge of $\overline{\text{PWR}}$ coincides with the presence of valid data on the local bus.
38	READY	I	<b>READY:</b> This pin is set high to insert wait states during a DMA transfer. The SNIC will sample this signal at t3 during DMA transfers.

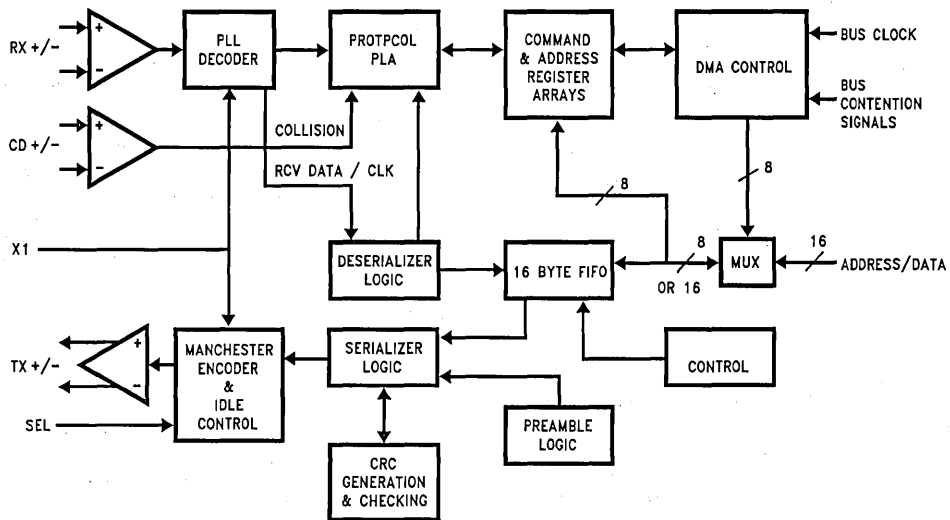
## Pin Description (Continued)

Pin No	Pin Name	I/O	Description
<b>BUS INTERFACE PINS (Continued)</b>			
39	PRQ/ADS1	O, Z	<p><b>PORT REQUEST/ADDRESS STROBE 1</b></p> <ul style="list-style-type: none"> <li>• <b>32-BIT MODE:</b> If LAS is set in the Data Configuration Register, this line is programmed as ADS1. It is used to strobe addresses A16–A31 into external latches. (A16–A31 are the fixed addresses stored in RSAR0, RSAR1). ADS1 will remain at TRI-STATE until BACK is received.</li> <li>• <b>16-BIT MODE:</b> If LAS is not set in the Data Configuration Register, this line is programmed as PRQ and is used for Remote DMA Transfers. The SNIC initiates a single remote DMA read or write operation by asserting this pin. In this mode PRQ will be a standard logic output.</li> </ul> <p><b>Note:</b> This line will power up as TRI-STATE until the Data Configuration Register is programmed.</p>
40	BACK	I	<p><b>BUS ACKNOWLEDGE:</b> Bus Acknowledge is an active high signal indicating that the CPU has granted the bus to the SNIC. If immediate bus access is desired, BREQ should be tied to BACK. <b>Tying BACK to V<sub>CC</sub> will result in a deadlock.</b></p>
41	BREQ	O	<p><b>BUS REQUEST:</b> Bus Request is an active high signal used to request the bus for DMA transfers. This signal is automatically generated when the FIFO needs servicing.</p>
65	RESET	I	<p><b>RESET:</b> Reset is active low and places the SNIC in a reset immediately, no packets are transmitted or received by the SNIC until STA bit is set. Affects Command Register, Interrupt Mask Register, Data Configuration Register and Transmit Configuration Register. The SNIC will execute reset within 10 BSCK cycles and TXC cycles.</p>
67	INT	O	<p><b>INTERRUPT:</b> Indicates that the SNIC requires CPU attention after reception transmission or completion of DMA transfers. The interrupt is cleared by writing to the ISR (Interrupt Service Register). All interrupts are maskable.</p>
68	WACK	I	<p><b>WRITE ACKNOWLEDGE:</b> Issued from system to SNIC to indicate that data has been written to the external latch. The SNIC will begin a write cycle to place the data in local memory.</p>
<b>NETWORK INTERFACE PINS</b>			
42, 43	TX– TX+	O	<p><b>TRANSMIT OUTPUT:</b> Differential driver which sends the encoded data to the transceiver. The outputs are source followers which require 270Ω pulldown resistors.</p>
46	TEST	I	<p><b>FACTORY TEST INPUT:</b> Used to check the chip's internal functions. Tied low during normal operation.</p>
47	SEL	I	<p><b>MODE SELECT:</b> When high, Transmit+ and Transmit– are the same voltage in the idle state. When low, Transmit+ is positive with respect to Transmit– in the idle state, at the transformer's primary.</p>
50	X1	I	<p><b>EXTERNAL OSCILLATOR INPUT</b></p>
51	GND/X2	O	<p><b>GROUND/X2:</b> This in should normally be connected to ground. It is possible to use a crystal oscillator using X1 and GND/X2 if certain precautions are taken. Contact National Semiconductor for more information.</p>
56	SNISEL	I	<p><b>FACTORY TEST INPUT:</b> For normal operation tied to V<sub>CC</sub>. When low enables the ENDEC module to be tested independently of the SNIC module.</p>
60, 61	RX– RX+	I	<p><b>RECEIVE INPUT:</b> Differential receive input pair from the transceiver.</p>
62, 63	CD– CD+	I	<p><b>COLLISION INPUT:</b> Differential collision pair input from the transceiver.</p>

**Pin Description** (Continued)

Pin No	Pin Name	I/O	Description
<b>POWER SUPPLY PINS</b>			
21, 48, 53, 55	V <sub>CC</sub>		<b>DIGITAL POSITIVE 5V SUPPLY PINS:</b>
20, 33, 49, 54, 66	GND		<b>DIGITAL NEGATIVE (GROUND) SUPPLY PINS:</b> It is suggested that a decoupling capacitor be connected between the V <sub>CC</sub> and GND pins.
59	V <sub>CC</sub>		<b>AUI RECEIVE 5V SUPPLY:</b> Power pin supplies 5V to the AUI receiver.
64	GND		<b>AUI RECEIVE GROUND:</b> Ground pin for AUI receiver.
45	V <sub>CC</sub>		<b>AUI TRANSMIT 5V SUPPLY:</b> Power pin supplies 5V to the AUI transmitter.
44	GND		<b>AUI TRANSMIT GROUND:</b> Ground pin for AUI transmitter
58	V <sub>CC</sub>		<b>VCO 5V SUPPLY:</b> Care should be taken to reduce noise on this pin as it supplies 5V to the ENDEC's Phase Lock Loop.
57	GND		<b>VCO GROUND PIN:</b> Care should be taken to reduce noise on this pin as it is the ground to the ENDEC's Phase Lock Loop.
<b>NO CONNECTION</b>			
1, 18, 35, 52	NC		<b>NO CONNECTION:</b> Do not connect to these pins.

**3.0 Block Diagram**



**FIGURE 1**

TL/F/10469-3

## 4.0 Functional Description (Refer to Figure 1)

### ENCODER/DECODER (ENDEC) MODULE

The ENDEC consists of four main logical blocks:

- a) The Manchester encoder accepts NRZ data from the controller, encodes the data to Manchester, and transmits it differentially to the transceiver, through the differential transmit driver.
- b) The Manchester decoder receives Manchester data from the transceiver, converts it to NRZ data and clock pulses, and sends it to the controller.
- c) The collision translator indicates to the controller the presence of a valid 10 MHz collision signal to the PLL.

#### MANCHESTER ENCODER AND DIFFERENTIAL DRIVER

The differential transmit pair, on the secondary of the employed transformer, drives up to 50 meters of twisted pair AUI cable. These outputs are source followers which require two 270 $\Omega$  pull-down resistors to ground.

The DP83901A allows both half-step and full-step to be compatible with Ethernet and IEEE 802.3. With the SEL pin low (for Ethernet I), Transmit+ is positive with respect to Transmit- during idle; with SEL high (for IEEE 802.3), Transmit+ and Transmit- are equal in the idle state. This provides zero differential voltage to operate with transformer-coupled loads.

#### MANCHESTER DECODER

The decoder consists of a differential receiver and a PLL to separate a Manchester decoded data stream into internal clock signals and data. The differential input must be externally terminated with two 39 $\Omega$  resistors connected in series if the standard 78 $\Omega$  transceiver drop cable is used, in thin Ethernet applications, these resistors are optional. To prevent noise from falsely triggering the decoder, a squelch circuit at the input rejects signals with levels less than -175 mV. Signals more negative than -300 mV and a duration greater than 30 ns are decoded. Data becomes valid typically within 5 bit times. The DP83901A may tolerate bit jitter up to 18 ns in the received data. The decoder detects the end of a frame when no more mid-bit transitions are detected.

#### COLLISION TRANSLATOR

When the Ethernet transceiver (DP8392 CTI) detects a collision, it generates a 10 MHz signal to the differential collision inputs (CD $\pm$ ) of the DP83901A. When these inputs are detected active, the DP83901A uses this signal to back off its current transmission and reschedule another one.

The collision differential inputs are terminated the same way as the differential receive inputs. The squelch circuitry is also similar, rejecting pulses with levels less than -175 mV.

### NIC (Media Access Control) MODULE

#### RECEIVE DESERIALIZER

The Receive Deserializer is activated when the input signal Carrier Sense is asserted to allow incoming bits to be shifted into the shift register by the receive clock. The serial receive data is also routed to the CRC generator/checker. The Receive Deserializer includes a synch detector which detects the SFD (Start of Frame Delimiter) to establish where byte boundaries within the serial bit stream are located. After every eight receive clocks, the byte wide data is transferred to the 16-byte FIFO and the Receive Byte Count is incremented. The first six bytes after the SFD are checked for valid comparison by the Address Recognition Logic. If the Address Recognition Logic does not recognize the packet, the FIFO is cleared.

#### CRC GENERATOR/CHECKER

During transmission, the CRC logic generates a local CRC field for the transmitted bit sequence. The CRC encodes all fields after the SFD. The CRC is shifted out MSB first following the last transmit byte. During reception the CRC logic generates a CRC field from the incoming packet. This local CRC is serially compared to the incoming CRC appended to the end of the packet by the transmitting node. If the local and received CRC match, a specific pattern will be generated and decoded to indicate no data errors. Transmission errors result in different pattern and are detected, resulting in rejection of a packet (if so programmed).

#### TRANSMIT SERIALIZER

The Transmit Serializer reads parallel data from the FIFO and serializes it for transmission. The serializer is clocked by the transmit clock generated internally. The serial data is also shifted into the CRC generator/checker. At the beginning of each transmission, the Preamble and Synch Generator append 62 bits of 1,0 preamble and a 1,1 synch pattern. After the last data byte of the packet has been serialized the 32-bit FCS field is shifted directly out of the CRC generator. In the event of a collision the Preamble and Synch generator is used to generate a 32-bit JAM pattern of all 1's.

#### ADDRESS RECOGNITION LOGIC

The address recognition logic compares the Destination Address Field (first 6 bytes of the received packet) to the Physical address registers stored in the Address Register Array. If any one of the six bytes does not match the pre-programmed physical address, the Protocol Control Logic rejects the packet. All multicast destination addresses are filtered using a hashing technique. (See register description.) If the multicast address indexes a bit that has been set in the filter bit array of the Multicast Address Register Array the packet is accepted, otherwise it is rejected by the Proto-

## 4.0 Functional Description (Continued)

col Control Logic. Each destination address is also checked for all 1's which is the reserved broadcast address.

### FIFO AND BUS OPERATIONS

#### Overview

To accommodate the different rates at which data comes from (or goes to) the network and goes to (or comes from) the system memory, the SNIC contains a 16-byte FIFO for buffering data between the media. The FIFO threshold is programmable, allowing filling (or emptying) the FIFO at different rates. When the FIFO has filled to its programmed threshold, the local DMA channel transfers these bytes (or words) into local memory. It is crucial that the local DMA is given access to the bus within a minimum bus latency time; otherwise a FIFO underrun (or overrun) occurs.

FIFO underruns or overruns are caused by two conditions: (1) the bus latency is so long that the FIFO has filled (or emptied) from the network before the local DMA has serviced the FIFO and (2) the bus latency has slowed the throughput of the local DMA to point where it is slower than the network data rate (10 Mbit/sec). This second condition is also dependent upon DMA clock and word width (byte wide or word wide). The worst case condition ultimately limits the overall bus latency which the SNIC can tolerate.

#### Beginning of Receive

At the beginning or reception, the SNIC stores entire Address field of each incoming packet in the FIFO to determine whether the packet matches its Physical Address Registers or maps to one of its Multicast Registers. This causes the FIFO to accumulate 8 bytes. Furthermore, there are some synchronization delays in the DMA PLA. Thus, the actual time that BREQ is asserted from the time the Start of Frame Delimiter (SFD) is detected is 7.8  $\mu$ s. This operation affects the bus latencies at 2 and 4-byte thresholds during the first receive BREQ since the FIFO must be filled to 8 bytes (or 4 words) before issuing a BREQ.

#### End of Receive

When the end of a packet is detected by the ENDEC module, the SNIC enters its end of packet processing sequence, emptying its FIFO and writing the status information at the beginning of the packet. The SNIC holds onto the bus for the entire sequence. The longest time BREQ may be extended occurs when a packet ends just as the SNIC performs its last FIFO burst. The SNIC, in this case, performs a programmed burst transfer followed by flushing the remaining bytes in the FIFO, and completed by writing the header information to memory. The following steps occur during this sequence.

1. SNIC issues BREQ because the FIFO threshold has been reached.

2. During the burst, packet ends, resulting in BREQ extended.
3. SNIC flushes remaining bytes from FIFO.
4. SNIC performs internal processing to prepare for writing the header.
5. SNIC writes 4-byte (2-word) header.
6. SNIC de-asserts BREQ.

#### FIFO Threshold Detection

To assure that no overwriting of data in the FIFO, the FIFO logic flags a FIFO overrun as the 13th byte is written into the FIFO, effectively shortening the FIFO to 13 bytes. The FIFO logic also operates differently in Byte Mode and in Word Mode. In Byte Mode, a threshold is indicated when the  $n+1$  byte has entered the FIFO; thus, with an 8-byte threshold, the SNIC issues Bus Request (BREQ) when the 9th byte has entered the FIFO. For Word Mode, BREQ is not generated until the  $n+2$  bytes have entered the FIFO. Thus, with a 4 word threshold (equivalent to 8-byte threshold), BREQ is issued when the 10th byte has entered the FIFO.

#### Beginning of Transmit

Before transmitting, the SNIC performs a prefetch from memory to load the FIFO. The number of bytes prefetched is the programmed FIFO threshold. The next BREQ is not issued until after the SNIC actually begins transmitting data, i.e., after SFD.

#### Reading the FIFO

During normal operation, the FIFO must not be read. The SNIC will not issue an ACKnowledge back to the CPU if the FIFO is read. The FIFO should only be read during loopback diagnostics.

#### PROTOCOL PLA

The protocol PLA is responsible for implementing the IEEE 802.3 protocol, including collision recovery with random backoff. The Protocol PLA also formats packets during transmission and strips preamble and synch during reception.

#### DMA AND BUFFER CONTROL LOGIC

The DMA and Buffer Control Logic is used to control two 16-bit DMA channels. During reception, the local DMA stores packets in a receive buffer ring, located in buffer memory. During transmission the Local DMA uses programmed pointer and length registers to transfer a packet from local buffer memory to the FIFO. A second DMA channel is used as a slave DMA to transfer data between the local buffer memory and the host system. The Local DMA and Remote DMA are internally arbitrated, with the Local DMA channel having highest priority. Both DMA channels use a common external bus clock to generate all required bus timing. External arbitration is performed with a standard bus request, bus acknowledge handshake protocol.

## 5.0 Transmit/Receive Packet Encapsulation/Decapsulation

A standard IEEE 802.3 packet consists of the following fields: preamble, Start of Frame Delimiter (SFD), destination address, source address, length, data, and Frame Check Sequence (FCS). The typical format is shown in *Figure 2*. The packets are Manchester encoded and decoded by the ENDEC module and transferred serially to the NIC module using NRZ data with a clock. All fields are of fixed length except for the data field. The SNIC generates and appends the preamble, SFD and FCS field during transmission. The Preamble and SFD fields are stripped during reception. (The CRC is passed through to buffer memory during reception.)

### PREAMBLE AND START OF FRAME DELIMITER (SFD)

The Manchester encoded alternating 1,0 preamble field is used by the ENDEC to acquire bit synchronization with an incoming packet. When transmitted each packet contains 62 bits of alternating 1,0 preamble. Some of this preamble will be lost as the packet travels through the network. The preamble field is stripped by the NIC module. Byte alignment is performed with the Start of Frame Delimiter (SFD) pattern which consists of two consecutive 1's. The SNIC does not treat the SFD pattern as a byte, it detects only the two bit pattern. This allows any preceding preamble within the SFD to be used for phase locking.

### DESTINATION ADDRESS

The destination address indicates the destination of the packet on the network and is used to filter unwanted packets from reaching a node. There are three types of address formats supported by the SNIC: physical, multicast and broadcast. The physical address is a unique address that corresponds only to a single node. All physical addresses have an MSB of "0". These addresses are compared to the internally stored physical address registers. Each bit in the destination address must match in order for the SNIC to accept the packet. Multicast addresses begin with an MSB

of "1". The SNIC filters multicast addresses using a standard hashing algorithm that maps all multicast addresses into a 6-bit value. This 6-bit value indexes a 64-bit array that filters the value. If the address consists of all 1's it is a broadcast address, indicating that the packet is intended for all nodes. A promiscuous mode allows reception of all packets: the destination address is not required to match any filters. Physical, broadcast, multicast, and promiscuous address modes can be selected.

### SOURCE ADDRESS

The source address is the physical address of the node that sent the packet. Source addresses cannot be multicast or broadcast addresses. This field is simply passed to buffer memory.

### LENGTH FIELD

The 2-byte length field indicates the number of bytes that are contained in the data field of the packet. This field is not interpreted by the SNIC.

### DATA FIELD

The data field consists of anywhere from 46 to 1500 bytes. Messages longer than 1500 bytes need to be broken into multiple packets. Messages shorter than 46 bytes will require appending a pad to bring the data field to the minimum length of 46 bytes. If the data field is padded, the number of valid data bytes is indicated in the length field. **The SNIC does not strip or append pad bytes for short packets, or check for oversize packets.**

### FCS FIELD

The Frame Check Sequence (FCS) is a 32-bit CRC field calculated and appended to a packet during transmission to allow detection of errors when a packet is received. During reception, error free packets result in a specific pattern in the CRC generator. Packets with improper CRC will be rejected. The AUTODIN II ( $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X^1 + 1$ ) polynomial is used for the CRC calculations.

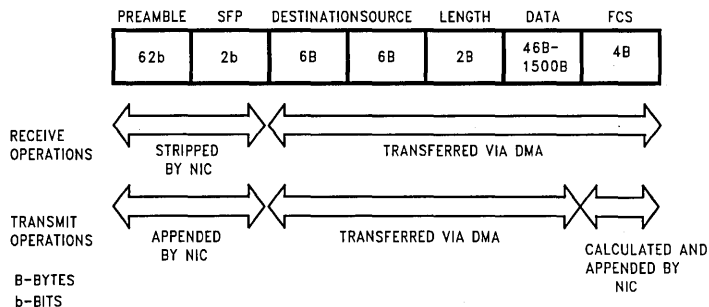


FIGURE 2

TL/F/10469-5

## 6.0 Direct Memory Access Control (DMA)

The DMA capabilities of the SNIC greatly simplify the use of the DP83901A in typical configurations. The local DMA channel transfers data between the FIFO and memory. On transmission, the packet is DMA'd from memory to the FIFO in bursts. Should a collision occur (up to 15 times), the packet is retransmitted with no processor intervention. On reception, packets are DMAed from the FIFO to the receive buffer ring (as explained below).

A remote DMA channel is also provided on the SNIC to accomplish transfers between a buffer memory and system memory. The two DMA channels can alternatively be combined to form a single 32-bit address with 8- or 16-bit data.

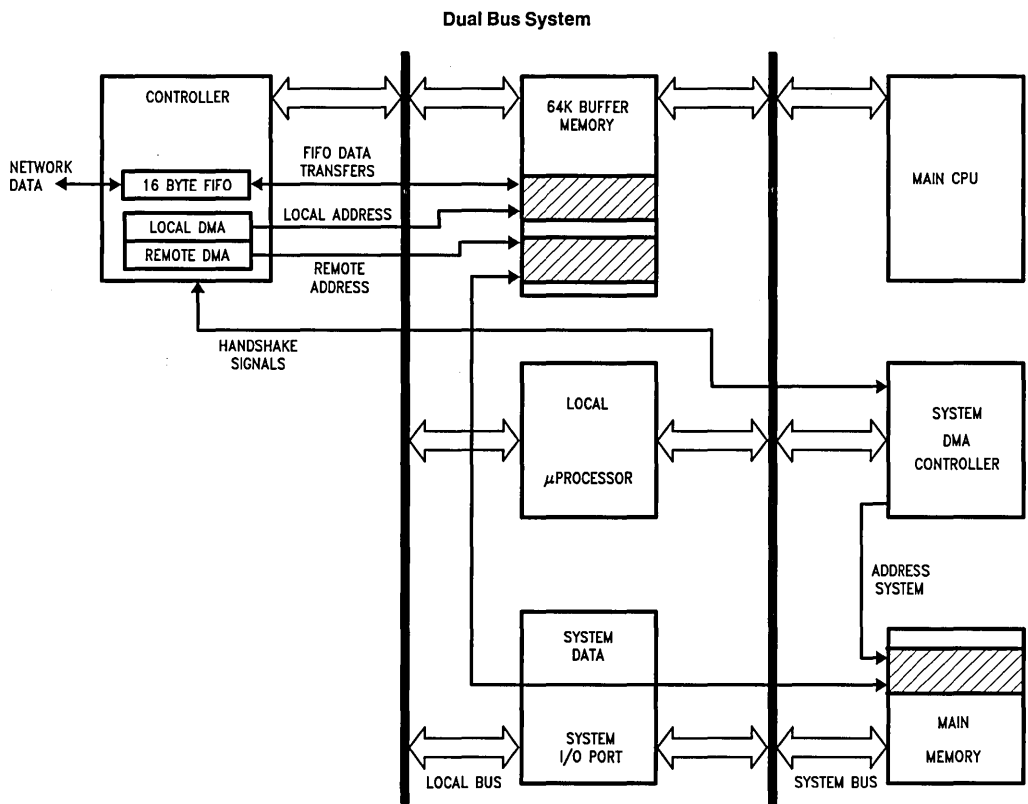
### DUAL DMA CONFIGURATION

An example configuration using both the local and remote DMA channels is shown below. Network activity is isolated

on a local bus, where the SNIC's local DMA channel performs burst transfers between the buffer memory and the SNIC's FIFO. The Remote DMA transfers data between the buffer memory and the host memory via a bidirectional I/O port. The Remote DMA provides local addressing capability and is used as a slave DMA by the host. Host side addressing must be provided by a host DMA or the CPU. The SNIC allows Local and Remote DMA operations to be interleaved.

### SINGLE CHANNEL DMA OPERATION

If desirable, the two DMA channels can be combined to provide a 32-bit DMA address. The upper 16 bits of the 32-bit address are static and are used to point to a 64 kbyte (or 32k word) page of memory where packets are to be received and transmitted.



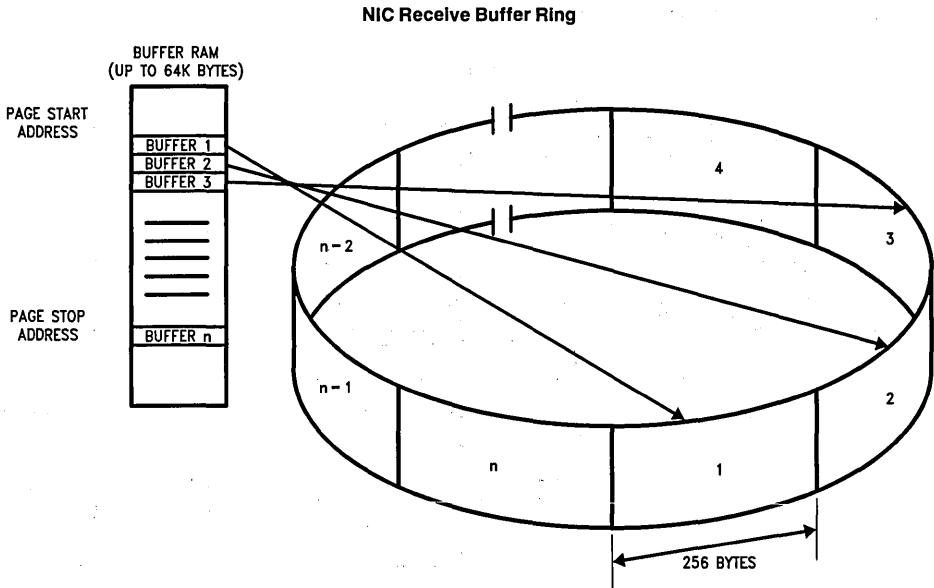
TL/F/10469-6

## 7.0 Packet Reception

The Local DMA receive channel uses a Buffer Ring Structure comprised of a series of contiguous fixed length 256-byte (128 word) buffers for storage of received packets. The location of the Receive Buffer Ring is programmed in two registers, a Page Start and a Page Stop Register. Ethernet packets consist of a distribution of shorter link control packets and longer data packets, the 256-byte buffer length provides a good compromise between short packets and longer packets to most efficiently use memory. In addition these buffers provide memory resources for storage of back-to-back packets in loaded networks. The assignment of buffers for storing packets is controlled by Buffer Management Logic in the SNIC. The Buffer Management Logic

provides three basic functions: linking receive buffers for long packets, recovery of buffers when a packet is rejected, and recirculation of buffer pages that have been read by the host.

At initialization, a portion of the 64 kbyte (or 32k word) address space is reserved for the receive buffer ring. Two 8-bit registers, The Page Start Address Register (PSTART) and the Page Stop Address Register (PSTOP) define the physical boundaries of where the buffers reside. The SNIC treats the list of buffers as a logical ring; whenever the DMA address reaches the Page Stop Address, the DMA is reset to the Page Start Address.



TL/F/10469-7



## 7.0 Packet Reception (Continued)

### INITIALIZATION OF THE BUFFER RING

Two static registers and two working registers control the operation of the Buffer Ring. These are the Page Start Register, Page Stop Register (both described previously), the Current Page Register and the Boundary Pointer Register. The Current Page Register points to the first buffer used to store a packet and is used to restore the DMA for writing status to the Buffer Ring or for restoring the DMA address in the event of a Runt packet, a CRC, or Frame Alignment error. The Boundary Register points to the first packet in the Ring not yet read by the host. If the local DMA address ever reaches the Boundary, reception is aborted. The Boundary Pointer is also used to initialize the Remote DMA for removing a packet and is advanced when a packet is removed. A

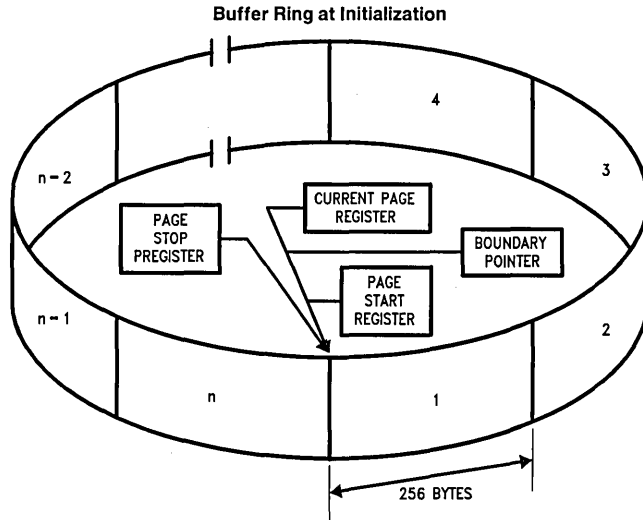
simple analogy to remember the function of these registers is that the Current Page Register acts as a Write Pointer and the Boundary Pointer acts as a Read Pointer.

**Note:** At initialization, the Page Start Register value should be loaded into both the Current Page Register and the Boundary Pointer Register.

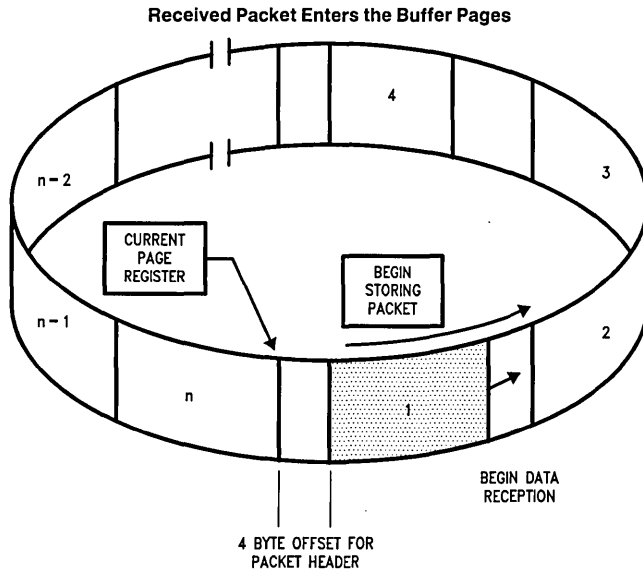
**Note:** The Page Start Register must not be initialized to 00H.

### BEGINNING OF RECEPTION

When the first packet begins arriving the SNIC begins storing the packet at the location pointed to by the Current Page Register. An offset of 4 bytes is saved in this first buffer to allow room for storing receive status corresponding to this packet.



TL/F/10469-8



TL/F/10469-9

## 7.0 Packet Reception (Continued)

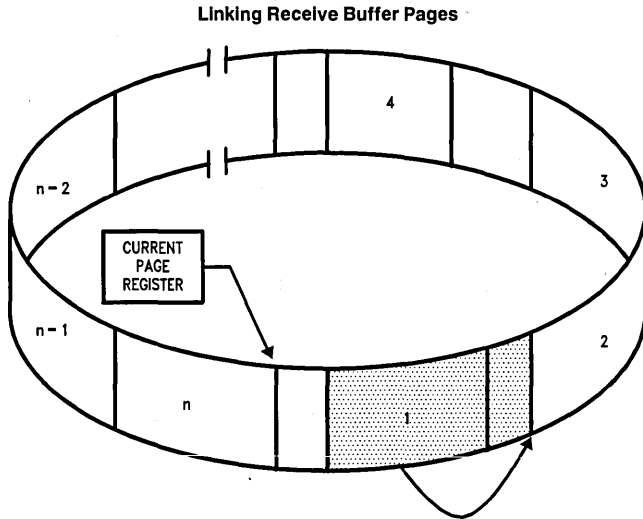
### LINKING RECEIVE BUFFER PAGES

If the length of the packet exhausts the first 256-byte buffer, the DMA performs a forward link to the next buffer to store the remainder of the packet. For a maximal length packet the buffer logic will link six buffers to store the entire packet. Buffers cannot be skipped when linking, a packet will always be stored in contiguous buffers. Before the next buffer can be linked, the Buffer Management Logic performs two comparisons. The first comparison tests for equality between the DMA address of the next buffer and the contents of the Page Stop Register. If the buffer address equals the Page Stop Register, the buffer management logic will restore the DMA to the first buffer in the Receive Buffer Ring value programmed in the Page Start Address Register. The second comparison tests for equality between the DMA ad-

dress of the next buffer address and the contents of the Boundary Pointer Register. If the two values are equal the reception is aborted. The Boundary Pointer Register can be used to protect against overwriting any area in the receive buffer ring that has not yet been read. When linking buffers, buffer management will never cross this pointer, effectively avoiding any overwrites. If the buffer address does not match either the Boundary Pointer or Page Stop Address, the link to the next buffer is performed.

### Linking Buffers

Before the DMA can enter the next contiguous 256-byte buffer, the address is checked for equality to PSTOP and to the Boundary Pointer. If neither are reached, the DMA is allowed to use the next buffer.



TL/F/10469-10

- 1) Check for = to PSTOP
- 2) Check for = to Boundary

## 7.0 Packet Reception (Continued)

### Buffer Ring Overflow

If the Buffer Ring has been filled and the DMA reaches the Boundary Pointer Address, reception of the incoming packet will be aborted by the SNIC. Thus, the packets previously received and still contained in the Ring will not be destroyed.

In heavily loaded network which cause overflows of the Receive Buffer Ring, the SNIC may disable the local DMA and suspend further receptions even if the Boundary register is advanced beyond the Current register. To guarantee this will not happen, a software reset must be issued during all Receive Buffer Ring overflows (indicated by the OVW bit in the Interrupt Status Register). The following procedure is required to recover from a Receiver Buffer Ring Overflow.

If this routine is not adhered to, the SNIC may act in an unpredictable manner. It should also be noted that it is not permissible to service an overflow interrupt by continuing to empty packets from the receive buffer without implementing the prescribed overflow routine. A flow chart of the SNIC's overflow routine can be found on the next page.

**Note:** It is necessary to define a variable in the driver, which will be called "Resend".

1. Read and store the value of the TXP bit in the SNIC's Command Register.
2. Issue the STOP command to the SNIC. This is accomplished by setting the STP bit in the SNIC's Command Register. Writing 21H to the Command Register will stop the SNIC.
3. Wait for at least 1.6 ms. Since the SNIC will complete any transmission or reception that is in progress, it is necessary to time out for the maximum possible duration of an Ethernet transmission or reception. By waiting 1.6 ms this is achieved with some guard band added. Previously, it was recommended that the RST bit of the Interrupt Status Register be polled to insure that the pending transmission or reception is completed. This bit is not a reliable indicator and subsequently should be ignored.
4. Clear the SNIC's Remote Byte Count registers (RBCR0 and RBCR1).
5. Read the stored value of the TXP bit from step 1, above. If this value is a 0, set the "Resend" variable to a 0 and jump to step 6. If this value is a 1, read the SNIC's Interrupt Status Register. If either the Packet Transmitted bit (PTX) or Trans-

mit Error bit (TXE) is set to a 1, set the "Resend" variable to a 0 and jump to step 6. If neither of these bits is set, place a 1 in the "Resend" variable and jump to step 6.

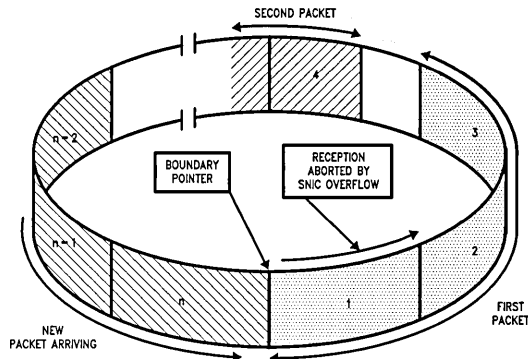
This step determines if there was a transmission in progress when the stop command was issued in step 2. If there was a transmission in progress, the SNIC's ISR is read to determine whether or not the packet was recognized by the SNIC. If neither the PTX nor TXE bit was set, then the packet will essentially be lost and re-transmitted only after a time-out takes place in the upper level software. By determining that the packet was lost at the driver level, a transmit command can be reissued to the SNIC once the overflow routine is completed (as in step 11). Also, it is possible for the SNIC to defer indefinitely, when it is stopped on a busy network. Step 5 also alleviates this problem. Step 5 is essential and should not be omitted from the overflow routine, in order for the SNIC to operate correctly.

6. Place the SNIC in either mode 1 or mode 2 loopback. This can be accomplished by setting bits D2 and D1, of the Transmit Configuration Register, to "0,1" or "1,0", respectively.
7. Issue the START command to the SNIC. This can be accomplished by writing 22H to the Command Register. This is necessary to activate the SNIC's Remote DMA channel.
8. Remove one or more packets from the receive buffer ring.
9. Reset the overwrite warning (OVW, overflow) bit in the Interrupt Status Register.
10. Take the SNIC out of loopback. This is done by writing the Transmit Configuration Register with the value it contains during normal operation. (Bits D2 and D1 should both be programmed to 0.)
11. If the "Resend" variable is set to a 1, reset the "Resend" variable and reissue the transmit command. This is done by writing a value of 26H to the Command Register. If the "Resend" variable is 0, nothing needs to be done.

**Note 1:** If Remote DMA is not being used, the SNIC does not need to be started before packets can be removed from the receive buffer ring. Hence, step 8 could be done before step 7.

**Note 2:** When the SNIC is in STOP mode, the Missed Talley Counter is disabled.

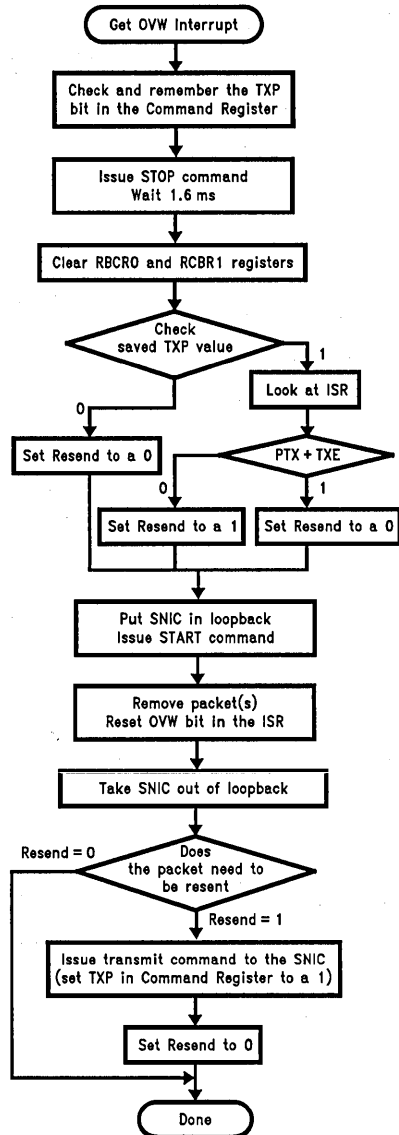
Received Packet Aborted If It Hits Boundary



TL/F/10469-11

## 7.0 Packet Reception (Continued)

Overflow Routine Flow Chart



TL/F/10469-52

## 7.0 Packet Reception (Continued)

### Enabling the SNIC On An Active Network

After the SNIC has been initialized the procedure for disabling and then re-enabling the SNIC on the network is similar to handling Receive Buffer Ring overflow as described previously.

1. Program Command Register for page 0 (Command Register = 21H)
2. Initialize Data Configuration Register (DCR)
3. Clear Remote Byte Count Registers (RBCR0, RBCR1)
4. Initialize Receive Configuration Register (RCR)
5. Place the SNIC in LOOPBACK mode 1 or 2 (Transmit Configuration Register = 02H or 04H)
6. Initialize Receive Buffer Ring: Boundary Pointer (BNDRY), Page Start (PSTART), and Page Stop (PSTOP)
7. Clear Interrupt Status Register (ISR) by writing 0FFH to it.
8. Initialize Interrupt Mask Register (IMR)
9. Program Command Register for page 1 (Command Register = 61H)
  - i) Initialize Physical Address Registers (PAR0-PAR5)
  - ii) Initialize Multicast Address Registers (MAR0-MAR7)
  - iii) Initialize CURRent pointer

- 10) Put SNIC in START mode (Command Register = 22H).  
The local receive DMA is still not active since the SNIC is in LOOPBACK.
- 11) Initialize the Transmit Configuration for the intended value. The SNIC is now ready for transmission and reception.

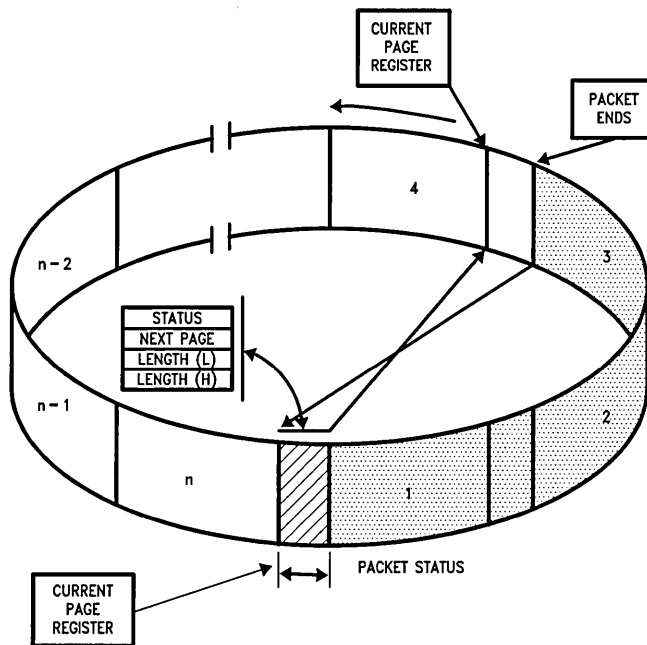
### END OF PACKET OPERATIONS

At the end of the packet the SNIC determines whether the received packet is to be accepted or rejected. It either branches to a routine to store the Buffer Header or to another routine that recovers the buffers used to store the packet.

### SUCCESSFUL RECEPTION

If the packet is successfully received, the DMA is restored to the first buffer used to store the packet (pointed to by the Current Page Register). The DMA then stores the Receive Status, a Pointer to where the next packet will be stored (Buffer 4) and the number of received bytes. Note that the remaining bytes in the last buffer are discarded and reception of the next packet begins on the next empty 256-byte buffer boundary. The Current Page Register is then initialized to the next available buffer in the Buffer Ring. (The location of the next buffer had been previously calculated and temporarily stored in an internal scratchpad register.)

Termination of Received Packet—Packet Accepted



TL/F/10469-12

## 7.0 Packet Reception (Continued)

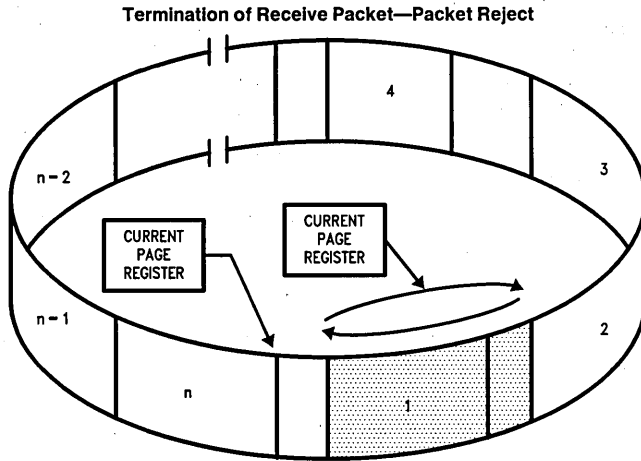
### BUFFER RECOVERY FOR REJECTED PACKETS

If the packet is a runt packet or contains CRC or Frame Alignment errors, it is rejected. The buffer management logic resets the DMA back to the first buffer page used to store the packet (pointed to by CURR), recovering all buffers that had been used to store the rejected packet. This operation will not be performed if the SNIC is programmed to accept either runt packets or packets with CRC or Frame Alignment

errors. The received CRC is always stored in buffer memory after the last byte of received data for the packet.

### Error Recovery

If the packet is rejected as shown, the DMA is restored by the SNIC by reprogramming the DMA starting address pointed to by the Current Page Register.



TL/F/10469-13

## 7.0 Packet Reception (Continued)

### REMOVING PACKETS FROM THE RING

Packets are removed from the ring using the Remote DMA or an external device. When using the Remote DMA the Send Packet command can be used. This programs the Remote DMA to automatically remove the received packet pointed to by the Boundary Pointer. At the end of the transfer, the SNIC moves the Boundary Pointer, freeing additional buffers for reception. The Boundary Pointer can also be moved manually by programming the Boundary Register.

### STORAGE FORMAT FOR RECEIVED PACKETS

The following diagrams describe the format for how received packets are placed into memory by the local DMA channel. These modes are selected in the Data Configuration Register.

AD15	AD8	AD7	AD0
Next Packet Pointer		Receive Status	
Receive Byte Count 1		Receive Byte Count 0	
Byte 2		Byte 1	

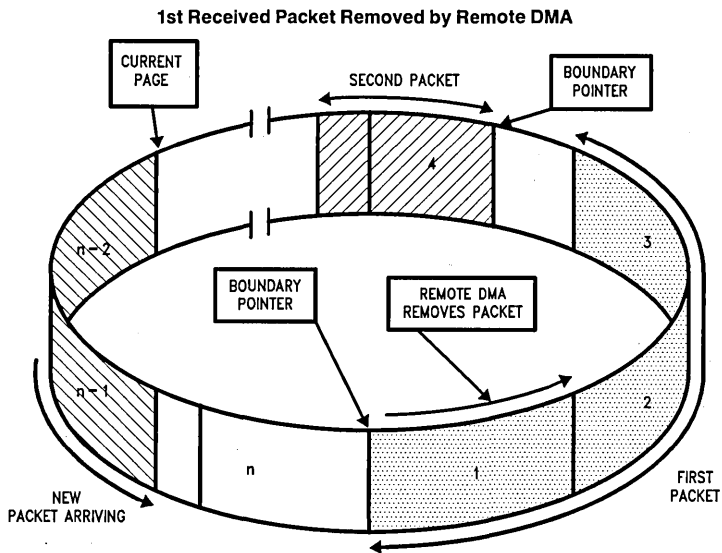
BOS = 0, WTS = 1 in Data Configuration Register. This format is used with Series 32xxx, or 808xx processors.

AD15	AD8	AD7	AD0
Next Packet Pointer		Receive Status	
Receive Byte Count 0		Receive Byte Count 1	
Byte 1		Byte 2	

BOS = 1, WTS = 1 in Data Configuration Register. This format is used with 680x0 type processors. (Note: The Receive Count ordering remains the same for BOS = 0 or 1.)

Receive Status
Next Packet Pointer
Receive Byte Count 0
Receive Byte Count 1
Byte 0
Byte 1

BOS = 0, WTS = 0 in Data Configuration Register. This format is used with general 8-bit processors.



TL/F/10469-14

## 8.0 Packet Transmission

The Local DMA is also used during transmission of a packet. Three registers control the DMA transfer during transmission, a Transmit Page Start Address Register (TPSR) and the Transmit Byte Count Registers (TBCRO, 1). When the SNIC receives a command to transmit the packet pointed to by these registers, buffer memory data will be moved into the FIFO as required during transmission. The SNIC will generate and append the preamble, synch and CRC fields.

General Transmit Packet Format		
Transmit	Destination Address	6 Bytes
Byte	Source Address	6 Bytes
Count	Type/Length	2 Bytes
TBCRO, 1	Data	≥ 46 Bytes
	Pad (If Data < 46 Bytes)	

### TRANSMIT PACKET ASSEMBLY

The SNIC requires a contiguous assembled packet with the format shown. The transmit byte count includes the Destination Address, Source Address, Length Field and Data. It does not include preamble and CRC. When transmitting data smaller than 46 bytes, the packet must be padded to a minimum size of 64 bytes. The programmer is responsible for adding and stripping pad bytes.

### TRANSMISSION

Prior to transmission, the TPSR (Transmit Page Start Register) and TBCRO, TBCR1 (Transmit Byte Count Registers) must be initialized. To initiate transmission of the packet the TXP bit in the Command Register is set. The Transmit Status Register (TSR) is cleared and the SNIC begins to prefetch transmit data from memory (unless the SNIC is currently receiving). If the interframe gap has timed out the SNIC will begin transmission.

### CONDITIONS REQUIRED TO BEGIN TRANSMISSION

In order to transmit a packet, the following three conditions must be met:

1. The Interframe Gap Timer has timed out the first 6.4 μs of the Interframe Gap.
2. At least one byte has entered the FIFO. (This indicates that the burst transfer has been started.)
3. If a collision had been detected then before transmission the packet time must have timed out.

In typical systems the SNIC prefetches the first burst of bytes before the 6.4 μs timer expires. The time during which SNIC transmits preamble can also be used to load the FIFO.

**Note:** If carrier sense is asserted before a byte has been loaded into the FIFO, the SNIC will become a receiver.

### COLLISION RECOVERY

During transmission, the Buffer Management logic monitors the transmit circuitry to determine if a collision has occurred. If a collision is detected, the Buffer Management logic will reset the FIFO and restore the Transmit DMA pointers for retransmission of the packet. The COL bit will be set in the TSR and the NCR (Number of Collisions Register) will be incremented. If 15 retransmissions each result in a collision the transmission will be aborted and the ABT bit in the TSR will be set.

**Note:** NCR reads as zeroes if excessive collisions are encountered.

### TRANSMIT PACKET ASSEMBLY FORMAT

The following diagrams describe the format for how packets must be assembled prior to transmission for different byte ordering schemes. The various formats are selected in the Data Configuration Register.

D15	D8	D7	D0
Destination Address 1		Destination Address 0	
Destination Address 3		Destination Address 2	
Destination Address 5		Destination Address 4	
Source Address 1		Source Address 0	
Source Address 3		Source Address 2	
Source Address 5		Source Address 4	
Type/Length 1		Type/Length 0	
Data 1		Data 0	

BOS = 0, WTS = 1 in Data Configuration Register.

This format is used with Series 32xxx, or 808xx processors.

D15	D8	D7	D0
Destination Address 0		Destination Address 1	
Destination Address 2		Destination Address 3	
Destination Address 4		Destination Address 5	
Source Address 0		Source Address 1	
Source Address 2		Source Address 3	
Source Address 4		Source Address 5	
Type/Length 0		Type/Length 1	
Data 0		Data 1	

BOS = 1, WTS = 1 in Data Configuration Register.

This format is used with 680x0 type processors.

D1	D0
Destination Address 0	
Destination Address 1	
Destination Address 2	
Destination Address 3	
Destination Address 4	
Destination Address 5	
Source Address 0	
Source Address 1	
Source Address 2	
Source Address 3	
Source Address 4	
Source Address 5	

BOS = 0, WTS = 0 in Data Configuration Register.

This format is used with general 8-bit processors.

**Note:** All examples above will result in a transmission of a packet in order of DA0, DA1, DA3 ... bits within each byte will be transmitted least significant bit first.

DA = Destination Address.



### 9.0 Remote DMA

The Remote DMA channel is used to both assemble packets for transmission, and to remove received packets from the Receive Buffer Ring. It may also be used as a general purpose slave DMA channel for moving blocks of data or commands between host memory and local buffer memory. There are three modes of operation, Remote Write, Remote Read, or Send Packet.

Two register pairs are used to control the Remote DMA, a Remote Start Address (RSAR0, RSAR1) and a Remote Byte Count (RBCR0, RBCR1) register pair. The Start Address Register pair points to the beginning of the block to be moved while the Byte Count Register pair is used to indicate the number of bytes to be transferred. Full handshake logic is provided to move data between local buffer memory and a bidirectional I/O port.

#### REMOTE WRITE

A Remote Write transfer is used to move a block of data from the host into local buffer memory. The Remote DMA will read data from the I/O port and sequentially write it to local buffer memory beginning at the Remote Start Address. The DMA Address will be incremented and the Byte Counter will be decremented after each transfer. The DMA is terminated when the Remote Byte Count Register reaches a count of zero.

#### REMOTE READ

A Remote Read transfer is used to move a block of data from local buffer memory to the host. The Remote DMA will

sequentially read data from the local buffer memory, beginning at the Remote Start Address, and write data to the I/O port. The DMA Address will be incremented and the Byte Counter will be decremented after each transfer. The DMA is terminated when the Remote Byte Count Register reaches zero.

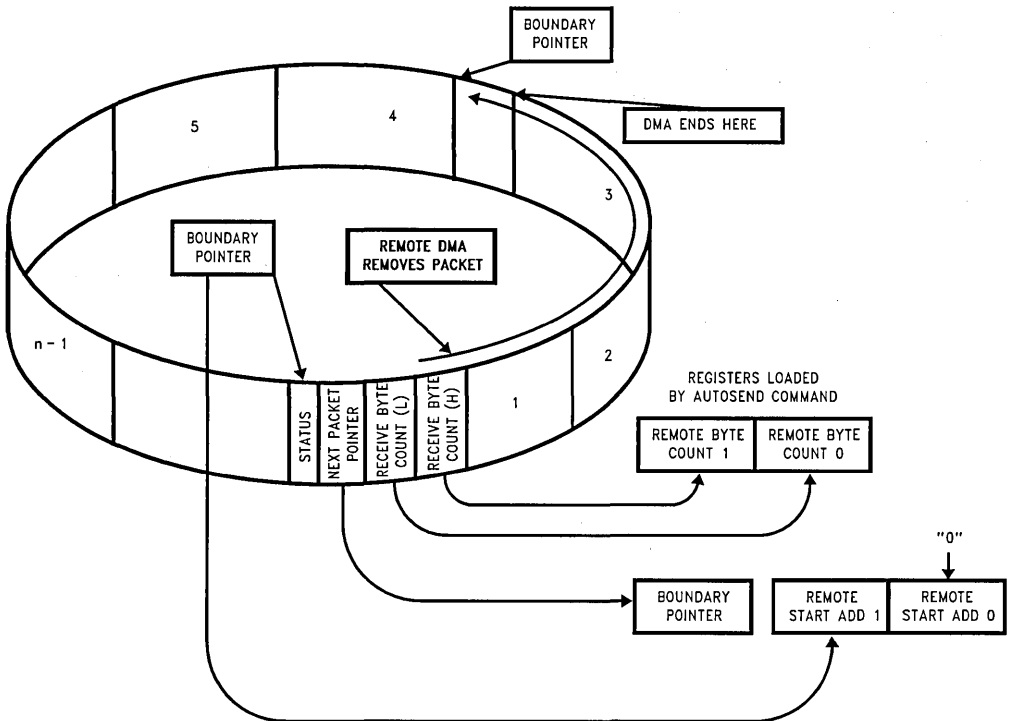
#### SEND PACKET COMMAND

The Remote DMA channel can be automatically initialized to transfer a single packet from the Receive Buffer Ring. The CPU begins this transfer by issuing a "Send Packet" Command. The DMA will be initialized to the value of the Boundary Pointer Register and the Remote Byte Count Register pair (RBCR0, RBCR1) will be initialized to the value of the Receive Byte Count fields found in the Buffer Header of each packet. After the data is transferred, the Boundary Pointer is advanced to allow the buffers to be used for new receive packets. The Remote Read will terminate when the Byte Count equals zero. The Remote DMA is then prepared to read the next packet from the Receive Buffer Ring. If the DMA pointer crosses the Page Stop Register, it is reset to the Page Start Address. This allows the Remote DMA to remove packets that have wrapped around to the top of the Receive Buffer Ring.

**Note 1:** In order for the SNIC to correctly execute the Send Packet Command, the upper Remote Byte Count Register (RBCR1) must first be loaded with 0FH.

**Note 2:** The Send Packet command cannot be used with 680x0 type processors.

Remote DMA Autoinitialization from Buffer Ring



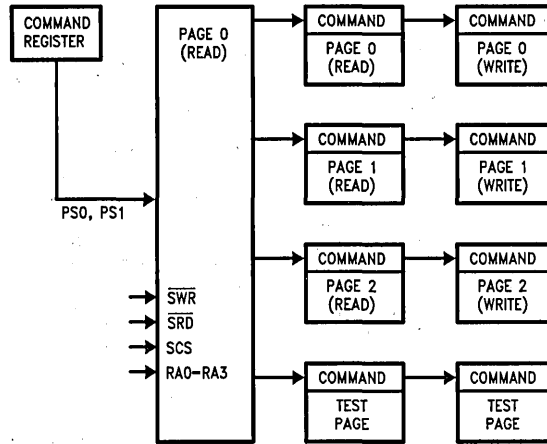
TL/F/10469-15

## 10.0 Internal Registers

All registers are 8-bit wide and mapped into four pages which are selected in the Command Register (PS0, PS1). Pins RA0-RA3 are used to address registers within each page. Page 0 registers are those registers which are com-

monly accessed during SNIC operation while page 1 registers are used primarily for initialization. The registers are partitioned to avoid having to perform two write/read cycles to access commonly used registers.

### 10.1 REGISTER ADDRESS MAPPING



TL/F/10469-16

### 10.2 REGISTER ADDRESS ASSIGNMENTS

Page 0 Address Assignments (PS1 = 0, PS0 = 0)

RA0-RA3	RD	WR
00H	Command (CR)	Command (CR)
01H	Current Local DMA Address 0 (CLDA0)	Page Start Register (PSTART)
02H	Current Local DMA Address 1 (CLDA1)	Page Stop Register (PSTOP)
03H	Boundary Pointer (BNRY)	Boundary Pointer (BNRY)
04H	Transmit Status Register (TSR)	Transmit Page Start Address (TPSR)
05H	Number of Collisions Register (NCR)	Transmit Byte Count Register 0 (TBCR0)
06H	FIFO (FIFO)	Transmit Byte Count Register 1 (TBCR1)
07H	Interrupt Status Register (ISR)	Interrupt Status Register (ISR)
08H	Current Remote DMA Address 0 (CRDA0)	Remote Start Address Register 0 (RSAR0)

RA0-RA3	RD	WR
09H	Current Remote DMA Address 1 (CRDA1)	Remote Start Address Register 1 (RSAR1)
0AH	Reserved	Remote Byte Count Register 0 (RBCR0)
0BH	Reserved	Remote Byte Count Register 1 (RBCR1)
0CH	Receive Status Register (RSR)	Receive Configuration Register (RCR)
0DH	Tally Counter 0 (Frame Alignment Errors) (CNTR0)	Transmit Configuration Register (TCR)
0EH	Tally Counter 1 (CRC Errors) (CNTR1)	Data Configuration Register (DCR)
0FH	Tally Counter 2 (Missed Packet Errors) (CNTR2)	Interrupt Mask Register (IMR)

## 10.0 Internal Registers (Continued)

Page 1 Address Assignments (PS1 = 0, PS0 = 1)

RA0-RA3	RD	WR
00H	Command (CR)	Command (CR)
01H	Physical Address Register 0 (PAR0)	Physical Address Register 0 (PAR0)
02H	Physical Address Register 1 (PAR1)	Physical Address Register 1 (PAR1)
03H	Physical Address Register 2 (PAR2)	Physical Address Register 2 (PAR2)
04H	Physical Address Register 3 (PAR3)	Physical Address Register 3 (PAR3)
05H	Physical Address Register 4 (PAR4)	Physical Address Register 4 (PAR4)
06H	Physical Address Register 5 (PAR5)	Physical Address Register 5 (PAR5)
07H	Current Page Register (CURR)	Current Page Register (CURR)
08H	Multicast Address Register 0 (MAR0)	Multicast Address Register 0 (MAR0)
09H	Multicast Address Register 1 (MAR1)	Multicast Address Register 1 (MAR1)
0AH	Multicast Address Register 2 (MAR2)	Multicast Address Register 2 (MAR2)
0BH	Multicast Address Register 3 (MAR3)	Multicast Address Register 3 (MAR3)
0CH	Multicast Address Register 4 (MAR4)	Multicast Address Register 4 (MAR4)
0DH	Multicast Address Register 5 (MAR5)	Multicast Address Register 5 (MAR5)
0EH	Multicast Address Register 6 (MAR6)	Multicast Address Register 6 (MAR6)
0FH	Multicast Address Register 7 (MAR7)	Multicast Address Register 7 (MAR7)

Page 2 Address Assignments (PS1 = 1, PS0 = 0)

RA0-RA3	RD	WR
00H	Command (CR)	Command (CR)
01H	Page Start Register (PSTART)	Current Local DMA Address 0 (CLDA0)
02H	Page Stop Register (PSTOP)	Current Local DMA Address 1 (CLDA1)
03H	Remote Next Packet Pointer	Remote Next Packet Pointer
04H	Transmit Page Start Address (TPSR)	Reserved
05H	Local Next Packet Pointer	Local Next Packet Pointer
06H	Address Counter (Upper)	Address Counter (Upper)
07H	Address Counter (Lower)	Address Counter (Lower)
08H	Reserved	Reserved
09H	Reserved	Reserved
0AH	Reserved	Reserved
0BH	Reserved	Reserved
0CH	Receive Configuration Register (RCR)	Reserved
0DH	Transmit Configuration Register (TCR)	Reserved
0EH	Data Configuration Register (DCR)	Reserved
0FH	Interrupt Mask Register (IMR)	Reserved

**Note:** Page 2 registers should only be accessed for diagnostic purposes. They should not be modified during normal operation.

Page 3 should never be modified.

## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS

#### COMMAND REGISTER (CR) 00H (READ/WRITE)

The Command Register is used to initiate transmissions, enable or disable Remote DMA operations and to select register pages. To issue a command the microprocessor sets the corresponding bit(s) (RD2, RD1, RD0, TXP). Further commands may be overlapped, but with the following rules: (1) If a transmit command overlaps with a remote DMA operation, bits RD0, RD1, and RD2 must be maintained for the remote DMA command when setting the TXP bit. Note, if a remote DMA command is re-issued when giving the transmit command, the DMA will complete immediately if the remote byte count register has not been reinitialized. (2) If a remote DMA operation overlaps a transmission, RD0, RD1, and RD2 may be written with the desired values and a "0" written to the TXP bit. Writing a "0" to this bit has no effect. (3) A remote write DMA may not overlap remote read operation or visa versa. Either of these operations must either complete or be aborted before the other operation may start. Bits PS1, PS0, RD2, and STP may be set any time.

7	6	5	4	3	2	1	0
PS1	PS0	RD2	RD1	RD0	TXP	STA	STP

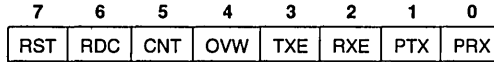
Bit	Symbol	Description																								
D0	STP	<p><b>Stop:</b> Software reset command, takes the controller offline, no packets will be received or transmitted. Any reception or transmission in progress will continue to completion before entering the reset state. To exit this state, the STP bit must be reset and the STA bit must be set high. To perform a software reset, this bit should be set high. The software reset has executed only when indicated by the RST bit in the ISR being set to 1. <b>STP powers up high.</b></p> <p><b>Note:</b> If the SNIC has previously been in start mode and the STP is set, both the STP and STA bits will remain set.</p>																								
D1	STA	<p><b>Start:</b> This bit is used to activate the SNIC after either power up, or when the SNIC has been placed in a reset mode by software command or error. <b>STA powers up low.</b></p>																								
D2	TXP	<p><b>Transmit Packet:</b> This bit must be set to initiate transmission of a packet. TXP is internally reset either after the transmission is completed or aborted. This bit should be set only after the Transmit Byte Count and Transmit Page Start registers have been programmed.</p>																								
D3, D4, and D5	RD0, RD1, and RD2	<p><b>Remote DMA Command:</b> These three encoded bits control operation of the Remote DMA channel. RD2 can be set to abort any Remote DMA command in progress. The Remote Byte Count Registers should be cleared when a Remote DMA has been aborted. The Remote Start Addresses are not restored to the starting address if the Remote DMA is aborted.</p> <table border="1"> <tr> <td>RD2</td> <td>RD1</td> <td>RD0</td> <td></td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>Not Allowed</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>Remote Read</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>Remote Write (Note 2)</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>Send Packet</td> </tr> <tr> <td>1</td> <td>X</td> <td>X</td> <td>Abort/Complete Remote DMA (Note 1)</td> </tr> </table> <p><b>Note 1:</b> If a remote DMA operation is aborted and the remote byte count has not decremented to zero, PRQ will remain high. A read acknowledge (RACK) on a write acknowledge (WACK) will reset PRQ low.</p> <p><b>Note 2:</b> For proper operation of the Remote Write DMA, there are two steps which must be performed before using the Remote Write DMA. The steps are as follows:</p> <ol style="list-style-type: none"> <li>I) Write a non-zero value into RBCR0.</li> <li>II) Set bits RD2, RD1, and RD0 to 0, 0, and 1.</li> <li>III) Issue the Remote Write DMA Command (RD2, RD1, RD0 = 0, 1, 0).</li> </ol>	RD2	RD1	RD0		0	0	0	Not Allowed	0	0	1	Remote Read	0	1	0	Remote Write (Note 2)	0	1	1	Send Packet	1	X	X	Abort/Complete Remote DMA (Note 1)
RD2	RD1	RD0																								
0	0	0	Not Allowed																							
0	0	1	Remote Read																							
0	1	0	Remote Write (Note 2)																							
0	1	1	Send Packet																							
1	X	X	Abort/Complete Remote DMA (Note 1)																							
D6 and D7	PS0 and PS1	<p><b>Page Select:</b> These two encoded bits select which register page is to be accessed with addresses RA0-3.</p> <table border="1"> <tr> <td>PS1</td> <td>PS0</td> <td></td> </tr> <tr> <td>0</td> <td>0</td> <td>Register Page 0</td> </tr> <tr> <td>0</td> <td>1</td> <td>Register Page 1</td> </tr> <tr> <td>1</td> <td>0</td> <td>Register Page 2</td> </tr> <tr> <td>1</td> <td>1</td> <td>Reserved</td> </tr> </table>	PS1	PS0		0	0	Register Page 0	0	1	Register Page 1	1	0	Register Page 2	1	1	Reserved									
PS1	PS0																									
0	0	Register Page 0																								
0	1	Register Page 1																								
1	0	Register Page 2																								
1	1	Reserved																								

## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS (Continued)

#### INTERRUPT STATUS REGISTER (ISR) 07H (READ/WRITE)

This register is accessed by the host processor to determine the cause of an interrupt. Any interrupt can be masked in the Interrupt Mask Register (IMR). Individual interrupt bits are cleared by writing a "1" into the corresponding bit of the ISR. The INT signal is active as long as any unmasked signal is set, and will not go low until all unmasked bits in this register have been cleared. The ISR must be cleared after power up by writing it with all 1's.



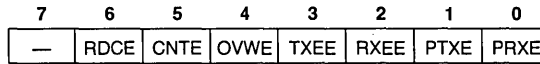
Bit	Symbol	Description
D0	PRX	<b>Packet Received:</b> Indicates packet received with no errors.
D1	PTX	<b>Packet Transmitted:</b> Indicates packet transmitted with no errors.
D2	RXE	<b>Receive Error:</b> Indicates that a packet was received with one or more of the following errors: — CRC Error — Frame Alignment Error — FIFO Overrun — Missed Packet
D3	TXE	<b>Transmit Error:</b> Set when packet transmitted with one or more of the following errors: — Excessive Collisions — FIFO Underrun
D4	OVW	<b>Overwrite Warning:</b> Set when receive buffer ring storage resources have been exhausted. (Local DMA has reached Boundary Pointer)
D5	CNT	<b>Counter Overflow:</b> Set when MSB of one or more of the Network Tally Counters has been set.
D6	RDC	<b>Remote DMA Complete:</b> Set when Remote DMA operation has been completed.
D7	RST	<b>Reset Status:</b> Set when SNIC enters reset state and cleared when a Start Command is issued to the CR. This bit is also set when a Receive Buffer Ring overflow occurs and is cleared when one or more packets have been removed from the ring. Writing to this bit has no effect.  <b>Note:</b> This bit does not generate an interrupt, it is merely a status indicator.

## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS (Continued)

#### INTERRUPT MASK REGISTER (IMR) 0FH (WRITE)

The Interrupt Mask Register is used to mask interrupts. Each interrupt mask bit corresponds to a bit in the Interrupt Status Register (ISR). If an interrupt mask bit is set, an interrupt will be issued whenever the corresponding bit in the ISR is set. If any bit in the IMR is set low, an interrupt will not occur when the bit in the ISR is set. **The IMR powers up to all zeroes.**



Bit	Symbol	Description
D0	PRXE	<b>Packet Received Interrupt Enable</b> 0: Interrupt Disabled 1: Enables Interrupt when packet received
D1	PTXE	<b>Packet Transmitted Interrupt Enable</b> 0: Interrupt Disabled 1: Enables Interrupt when packet is transmitted
D2	RXEE	<b>Receive Error Interrupt Enable</b> 0: Interrupt Disabled 1: Enables Interrupt when packet received with error
D3	TXEE	<b>Transmit Error Interrupt Enable</b> 0: Interrupt Disabled 1: Enables Interrupt when packet transmission results in error
D4	OVWE	<b>Overwrite Warning Interrupt Enable</b> 0: Interrupt Disabled 1: Enables Interrupt when Buffer Management Logic lacks sufficient buffers to store incoming packet
D5	CNTE	<b>Counter Overflow Interrupt Enable</b> 0: Interrupt Disabled 1: Enables Interrupt when MSB of one or more of the Network Statistics counters has been set
D6	RDCE	<b>DMA Complete Interrupt Enable</b> 0: Interrupt Disabled 1: Enables Interrupt when Remote DMA transfer has been completed
D7	Reserved	Reserved

## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS (Continued)

#### DATA CONFIGURATION REGISTER (DCR) 0EH (WRITE)

This Register is used to program the SNIC for 8- or 16-bit memory interface, select byte ordering in 16-bit applications and establish FIFO thresholds. **The DCR must be initialized prior to loading the Remote Byte Count Registers. LAS is set on power up.**

7	6	5	4	3	2	1	0
—	FT1	FT0	ARM	LS	LAS	BOS	WTS

Bit	Symbol	Description																				
D0	WTS	<p><b>Word Transfer Select</b>                      0: Selects byte-wide DMA transfers                      1: Selects word-wide DMA transfers</p> <p>; WTS establishes byte or word transfers for both Remote and Local DMA transfers  <b>Note:</b> When word-wide mode is selected up to 32k words are addressable; A0 remains low.</p>																				
D1	BOS	<p><b>Byte Order Select</b>                      0: MS byte placed on AD15–AD8 and LS byte on AD7–AD0. (32xxx, 80x86)                      1: MS byte placed on AD7–AD0 and LS byte on AD15–A8. (680x0)</p> <p>: Ignored when WTS is low</p>																				
D2	LAS	<p><b>Long Address Select</b>                      0: Dual 16-bit DMA mode                      1: Single 32-bit DMA mode</p> <p>; When LAS is high, the contents of the Remote DMA registers RSAR0, 1 are issued as A16–A31 Power up high</p>																				
D3	LS	<p><b>Loopback Select</b>                      0: Loopback mode selected. Bits D1 and D2 of the TCR must also be programmed for Loopback operation                      1: Normal Operation</p>																				
D4	ARM	<p><b>Auto-Initialize Remote</b>                      0: Send Command not executed, all packets removed from Buffer Ring under program control                      1: Send Command executed, Remote DMA auto-initialized to remove packets from Buffer Ring</p> <p><b>Note:</b> Send Command cannot be used with 680x0 byte processors.</p>																				
D5 and D6	FT0 and FT1	<p><b>FIFO Threshold Select:</b> Encoded FIFO threshold. Establishes point at which bus is requested when filling or emptying the FIFO. During reception, the FIFO threshold indicates the number of bytes (or words) the FIFO has filled serially from the network before bus request (BREQ) is asserted.</p> <p><b>Note:</b> FIFO threshold setting determines the DMA burst length.</p> <p style="text-align: center;">Receive Thresholds</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th style="text-align: center;">FT1</th> <th style="text-align: center;">FT0</th> <th style="text-align: center;">Word Wide</th> <th style="text-align: center;">Byte Wide</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1 Word</td> <td style="text-align: center;">2 Bytes</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">2 Words</td> <td style="text-align: center;">4 Bytes</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">4 Words</td> <td style="text-align: center;">8 Bytes</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">6 Words</td> <td style="text-align: center;">12 Bytes</td> </tr> </tbody> </table> <p>During transmission, the FIFO threshold indicates the number of bytes (or words) the FIFO has filled from the Local DMA before BREQ is asserted. Thus, the transmission threshold is 13 bytes less the received threshold.</p>	FT1	FT0	Word Wide	Byte Wide	0	0	1 Word	2 Bytes	0	1	2 Words	4 Bytes	1	0	4 Words	8 Bytes	1	1	6 Words	12 Bytes
FT1	FT0	Word Wide	Byte Wide																			
0	0	1 Word	2 Bytes																			
0	1	2 Words	4 Bytes																			
1	0	4 Words	8 Bytes																			
1	1	6 Words	12 Bytes																			

## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS (Continued)

#### TRANSMIT CONFIGURATION REGISTER (TCR) 0DH (WRITE)

The transmit configuration establishes the actions of the transmitter section of the SNIC during transmission of a packet on the network. **LB1** and **LB0** which select loopback mode power up as 0.

7	6	5	4	3	2	1	0
—	—	—	OFST	ATD	LB1	LB0	CRC

Bit	Symbol	Description																				
D0	CRC	<p><b>Inhibit CRC</b></p> <p>0: CRC appended by transmitter                      1: CRC inhibited by transmitter</p> <p>In loopback mode CRC can be enabled or disabled to test the CRC logic</p>																				
D1 and D2	LB0 and LB1	<p><b>Encoded Loopback Control:</b> These encoded configuration bits set the type of loopback that is to be performed. Note that loopback in mode 2 places the ENDEC Module in loopback mode and that D3 of the DCR must be set to zero for loopback operation.</p> <table border="1"> <tr> <td></td> <td>LB1</td> <td>LB0</td> <td></td> </tr> <tr> <td>Mode 0</td> <td>0</td> <td>0</td> <td>Normal Operation (LPBK = 0)</td> </tr> <tr> <td>Mode 1</td> <td>0</td> <td>1</td> <td>Internal NIC Module Loopback (LPBK = 0)</td> </tr> <tr> <td>Mode 2</td> <td>1</td> <td>0</td> <td>Internal ENDEC Module Loopback (LPBK = 1)</td> </tr> <tr> <td>Mode 3</td> <td>1</td> <td>1</td> <td>External Loopback (LPBK = 0)</td> </tr> </table>		LB1	LB0		Mode 0	0	0	Normal Operation (LPBK = 0)	Mode 1	0	1	Internal NIC Module Loopback (LPBK = 0)	Mode 2	1	0	Internal ENDEC Module Loopback (LPBK = 1)	Mode 3	1	1	External Loopback (LPBK = 0)
	LB1	LB0																				
Mode 0	0	0	Normal Operation (LPBK = 0)																			
Mode 1	0	1	Internal NIC Module Loopback (LPBK = 0)																			
Mode 2	1	0	Internal ENDEC Module Loopback (LPBK = 1)																			
Mode 3	1	1	External Loopback (LPBK = 0)																			
D3	ATD	<p><b>Auto Transmit Disable:</b> This bit allows another station to disable the SNIC's transmitter by transmission of a particular multicast packet. The transmitter can be re-enabled by resetting this bit or by reception of a second particular multicast packet.</p> <p>1: Reception of multicast address hashing to bit 62 disables transmitter, reception of multicast address hashing to bit 63 enables transmitter.</p>																				
D4	OFST	<p><b>Collision Offset Enable:</b> This bit modifies the backoff algorithm to allow prioritization of nodes.</p> <p>0: Backoff Logic implements normal algorithm.                      1: Forces Backoff algorithm modification to 0 to <math>2^{\min(3+n, 10)}</math> slot times for first three collisions, then follows standard backoff. (For the first three collisions, the station has higher average backoff delay making a low priority mode.)</p>																				
D5	Reserved	Reserved																				
D6	Reserved	Reserved																				
D7	Reserved	Reserved																				



## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS (Continued)

#### TRANSMIT STATUS REGISTER (TSR) 04H (READ)

This register records events that occur on the media during transmission of a packet. It is cleared when the next transmission is initiated by the host. All bits remain low unless the event that corresponds to a particular bit occurs during transmission. Each transmission should be followed by a read of this register. The contents of this register are not specified until after the first transmission.

7	6	5	4	3	2	1	0
OWC	CDH	FU	CRS	ABT	COL	—	PTX

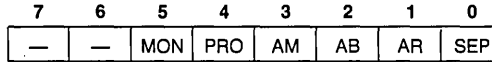
Bit	Symbol	Description
D0	PTX	<b>Packet Transmitted:</b> Indicates transmission without error. (No excessive collisions or FIFO underrun) (ABT = "0", FU = "0")
D1	Reserved	Reserved
D2	COL	<b>Transmit Collided:</b> Indicates that the transmission collided at least once with another station on the network. The number of collisions is recorded in the Number of Collisions Registers (NCR).
D3	ABT	<b>Transmit Aborted:</b> Indicates the SNIC aborted transmission because of excessive collisions. (Total number of transmissions including original transmission attempt equals 16.)
D4	CRS	<b>Carrier Sense Lost:</b> This bit is set when carrier is lost during transmission of the packet. Transmission is not aborted on loss of carrier.
D5	FU	<b>FIFO Underrun:</b> If the SNIC cannot gain access of the bus before the FIFO empties, this bit is set. Transmission of the packet will be aborted.
D6	CDH	<b>CD Heartbeat:</b> Failure of the transceiver to transmit a collision signal after transmission of a packet will set this bit. The Collision Detect (CD) heartbeat signal must commence during the first 6.4 $\mu$ s of the Interframe Gap following a transmission. In certain collisions, the CD Heartbeat bit will be set even though the transceiver is not performing the CD heartbeat test.
D7	OWC	<b>Out of Window Collision:</b> Indicates that a collision occurred after a slot time (51.2 $\mu$ s). Transmissions rescheduled as in normal collisions.

## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS (Continued)

#### RECEIVE CONFIGURATION REGISTER (RCR) OCH (WRITE)

This register determines operation of the SNIC during reception of a packet and is used to program what types of packets to accept.



Bit	Symbol	Description
D0	SEP	<p><b>Save Errored Packets</b></p> <p>0: Packets with receive errors are rejected.                      1: Packets with receive errors are accepted. Receive errors are CRC and Frame Alignment errors.</p>
D1	AR	<p><b>Accept Runt Packets:</b> This bit allows the receiver to accept packets that are smaller than 64 bytes. The packet must be at least 8 bytes long to be accepted as a runt.</p> <p>0: Packets with fewer than 64 bytes rejected.                      1: Packets with fewer than 64 bytes accepted.</p>
D2	AB	<p><b>Accept Broadcast:</b> Enables the receiver to accept a packet with an all 1's destination address.</p> <p>0: Packets with broadcast destination address rejected.                      1: Packets with broadcast destination address accepted.</p>
D3	AM	<p><b>Accept Multicast:</b> Enables the receiver to accept a packet with a multicast address, all multicast addresses must pass the hashing array.</p> <p>0: Packets with multicast destination address not checked.                      1: Packets with multicast destination address checked.</p>
D4	PRO	<p><b>Promiscuous Physical:</b> Enables the receiver to accept all packets with a physical address.</p> <p>0: Physical address of node must match the station address programmed in PAR0–PAR5.                      1: All packets with physical addresses accepted.</p>
D5	MON	<p><b>Monitor Mode:</b> Enables the receiver to check addresses and CRC on incoming packets without buffering to memory. The Missed Packet Tally counter will be incremented for each recognized packet.</p> <p>0: Packets buffered to memory.                      1: Packets checked for address match, good CRC and Frame Alignment but not buffered to memory.</p>
D6	Reserved	Reserved
D7	Reserved	Reserved

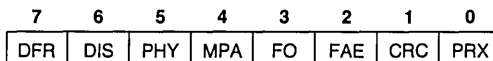
**Note:** D2 and D3 are "OR'd" together, i.e., if D2 and D3 are set the SNIC will accept broadcast and multicast addresses as well as its own physical address. To establish full promiscuous mode, bits D2, D3, and D4 should be set. In addition the multicast hashing array must be set to all 1's in order to accept all multicast addresses.

## 10.0 Internal Registers (Continued)

### 10.3 REGISTER DESCRIPTIONS (Continued)

#### RECEIVE STATUS REGISTER (RSR) 0CH (READ)

This register records status of the received packet, including information on errors and the type of address match, either physical or multicast. The contents of this register are written to buffer memory by the DMA after reception of a good packet. If packets with errors are to be saved the receive status is written to memory at the head of the erroneous packet if an erroneous packet is received. If packets with errors are to be rejected the RSR will not be written to memory. The contents will be cleared when the next packet arrives. CRC errors, Frame Alignment errors and missed packets are counted internally by the SNIC which relinquishes the Host from reading the RSR in real time to record errors for Network Management Functions. The contents of this register are not specified until after the first reception.



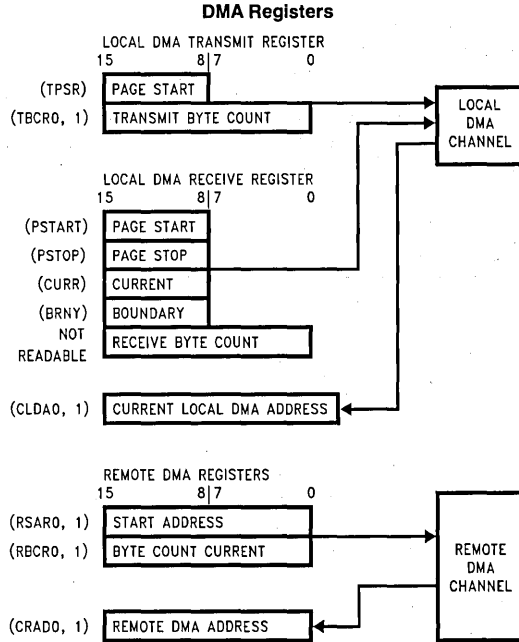
Bit	Symbol	Description
D0	PRX	<b>Packet Received Intact:</b> Indicates packet received without error. (Bits CRC, FAE, FO, and MPA are zero for the received packet.)
D1	CRC	<b>CRC Error:</b> Indicates packet received with CRC error. Increments Tally Counter (CNTR1). This bit will also be set for Frame Alignment errors.
D2	FAE	<b>Frame Alignment Error:</b> Indicates that the incoming packet did not end on a byte boundary and the CRC did not match at last byte boundary. Increments Tally Counter (CNTR0).
D3	FO	<b>FIFO Overrun:</b> This bit is set when the FIFO is not serviced causing overflow during reception. Reception of the packet will be aborted.
D4	MPA	<b>Missed Packet:</b> Set when a packet intended for node cannot be accepted by SNIC because of a lack of receive buffers or if the controller is in monitor mode and did not buffer the packet to memory. Increments Tally Counter (CNTR2).
D5	PHY	<b>Physical/Multicast Address:</b> Indicates whether received packet had a physical or multicast address type. 0: Physical Address Match 1: Multicast/Broadcast Address Match
D6	DIS	<b>Receiver Disabled:</b> Set when receiver disabled by entering Monitor mode. Reset when receiver is re-enabled when exiting Monitor mode.
D7	DFR	<b>Deferring:</b> Set when internal Carrier Sense or Collision signals are generated in the ENDEC module. If the transceiver has asserted the CD line as a result of the jabber, this bit will stay set indicating the jabber condition.

**Note:** Following coding applies to CRC and FAE bits.

FAE	CRC	Type of Error
0	0	No Error (Good CRC and <6 Dribble Bits)
0	1	CRC Error
1	0	Illegal, Will Not Occur
1	1	Frame Alignment Error and CRC Error

# 10.0 Internal Registers (Continued)

## 10.4 DMA REGISTERS



TL/F/10469-17

The DMA Registers are partitioned into groups; Transmit, Receive and Remote DMA Registers. The Transmit registers are used to initialize the Local DMA Channel for transmission of packets while the Receive Registers are used to initialize the Local DMA Channel for packet Reception. The Page Stop, Page Start, Current and Boundary Registers are used by the Buffer Management Logic to supervise the Receive Buffer Ring. The Remote DMA Registers are used to initialize the Remote DMA.

**Note:** In the figure above, registers are shown as 8 or 16 bits wide. Although some registers are 16-bit internal registers, all registers are accessed as 8-bit registers. Thus the 16-bit Transmit Byte Count Register is broken into two 8-bit registers, TBCR0, TBCR1. Also TPSR, PSTART, PSTOP, CURR and BNRY only check or control the upper 8 bits of address information on the bus. Thus, they are shifted to positions 15-8 in the diagram above.

### 10.5 TRANSMIT DMA REGISTERS

#### TRANSMIT PAGE START REGISTER (TPSR)

This register points to the assembled packet to be transmitted. Only the eight higher order addresses are specified since all transmit packets are assembled on 256-byte page boundaries. The bit assignment is shown below. The values placed in bits D7-D0 will be used to initialize the higher order address (A8-A15) of the Local DMA for transmission. The lower order bits (A7-A0) are initialized to zero.

Bit Assignment

	7	6	5	4	3	2	1	0
TPSR	A15	A14	A13	A12	A11	A10	A9	A8

(A7-A0 Initialized to Zero)

#### TRANSMIT BYTE COUNT REGISTER 0, 1 (TBCR0, TBCR1)

These two registers indicate the length of the packet to be transmitted in bytes. The count must include the number of bytes in the source, destination, length and data fields. The maximum number of transmit bytes allowed is 64 Kbytes. The SNIC will not truncate transmissions longer than 1500 bytes. The bit assignment is shown below:

	7	6	5	4	3	2	1	0
TBCR1	L15	L14	L13	L12	L11	L10	L9	L8

	7	6	5	4	3	2	1	0
TBCR0	L7	L6	L5	L4	L3	L2	L1	L0

## 10.0 Internal Registers (Continued)

### 10.6 LOCAL DMA RECEIVE REGISTERS

#### PAGE START AND STOP REGISTERS (PSTART, PSTOP)

The Page Start and Page Stop Registers program the starting and stopping address of the Receive Buffer Ring. Since the SNIC uses fixed 256-byte buffers aligned on page boundaries only the upper 8 bits of the start and stop address are specified.

#### PSTART, PSTOP Bit Assignment

	7	6	5	4	3	2	1	0
PSTART,	A15	A14	A13	A12	A11	A10	A9	A8
PSTOP								

#### BOUNDARY (BNRY) REGISTER

This register is used to prevent overflow of the Receive Buffer Ring. Buffer management compares the contents of this register to the next buffer address when linking buffers together. If the contents of this register match the next buffer address the Local DMA operation is aborted.

	7	6	5	4	3	2	1	0
BNRY	A15	A14	A13	A12	A11	A10	A9	A8

#### CURRENT PAGE REGISTER (CURR)

This register is used internally by the Buffer Management Logic as a backup register for reception. CURR contains the address of the first buffer to be used for a packet reception and is used to restore DMA pointers in the event of receive errors. This register is initialized to the same value as PSTART and should not be written to again unless the controller is Reset.

	7	6	5	4	3	2	1	0
CURR	A15	A14	A13	A12	A11	A10	A9	A8

#### CURRENT LOCAL DMA REGISTER 0,1 (CLDA0, 1)

These two registers can be accessed to determine the current local DMA address.

	7	6	5	4	3	2	1	0
CLDA1	A15	A14	A13	A12	A11	A10	A9	A8

	7	6	5	4	3	2	1	0
CLDA0	A7	A6	A5	A4	A3	A2	A1	A0

### 10.7 REMOTE DMA REGISTERS

#### REMOTE START ADDRESS REGISTERS (RSAR0, 1)

Remote DMA operations are programmed via the Remote Start Address (RSAR0, 1) and Remote Byte Count (RBCR0, 1) registers. The Remote Start Address is used to point to the start of the block of data to be transferred and the Remote Byte Count is used to indicate the length of the block (in bytes).

	7	6	5	4	3	2	1	0
RSAR1	A15	A14	A13	A12	A11	A10	A9	A8

	7	6	5	4	3	2	1	0
RSAR0	A7	A6	A5	A4	A3	A2	A1	A0

#### REMOTE BYTE COUNT REGISTERS (RBCR0, 1)

	7	6	5	4	3	2	1	0
RBCR1	BC15	BC14	BC13	BC12	BC11	BC10	BC9	BC8

	7	6	5	4	3	2	1	0
RBCR0	BC7	BC6	BC5	BC4	BC3	BC2	BC1	BC0

Note: RSAR0 programs the start address bits A0-A7.

RSAR1 programs the start address bits A8-A15.

Address incremented by two for word transfers, and by one for byte transfers. Byte Count decremented by two for word transfers and by one for byte transfers.

RBCR0 programs LSB byte count.

RBCR1 programs MSB byte count.

#### CURRENT REMOTE DMA ADDRESS (CRDA0, CRDA1)

The Current Remote DMA Registers contain the current address of the Remote DMA. The bit assignment is shown below:

	7	6	5	4	3	2	1	0
CRDA1	A15	A14	A13	A12	A11	A10	A9	A8

	7	6	5	4	3	2	1	0
CRDA0	A7	A6	A5	A4	A3	A2	A1	A0

### 10.8 PHYSICAL ADDRESS REGISTERS (PAR0-PAR5)

The physical address registers are used to compare the destination address of incoming packets for rejecting or accepting packets. Comparisons are performed on a byte-wide basis. The bit assignment shown below relates the sequence in PAR0-PAR5 to the bit sequence of the received packet.

	D7	D6	D5	D4	D3	D2	D1	D0
PAR0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
PAR1	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
PAR2	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
PAR3	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
PAR4	DA39	DA38	DA37	DA36	DA35	DA34	DA33	DA32
PAR5	DA47	DA46	DA45	DA44	DA43	DA42	DA41	DA40

#### Destination Address Source

P/S	DA0	DA1	DA2	DA3	...	DA46	DA47	SA0	...
-----	-----	-----	-----	-----	-----	------	------	-----	-----

Note: P/S = Preamble, Synch

DA0 = Physical/Multicast Bit

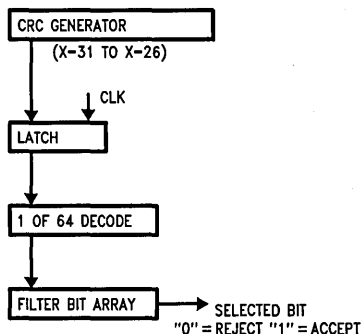
### 10.9 MULTICAST ADDRESS REGISTERS (MAR0-MAR7)

The multicast address registers provide filtering of multicast addresses hashed by the CRC logic. All destination addresses are fed through the CRC logic and as the last bit of the destination address enters the CRC, the 6 most signifi-

## 10.0 Internal Registers (Continued)

cant bits of the CRC generator are latched. These 6 bits are then decoded by a 1 of 64 decode to index a unique filter bit (FB0–63) in the multicast address registers. If the filter bit selected is set, the multicast packet is accepted. The system designer would use a program to determine which filter bits to set in the multicast registers. All multicast filter bits that correspond to multicast address accepted by the node are then set to one. To accept all multicast packets all of the registers are set to all ones.

**Note:** Although the hashing algorithm does not guarantee perfect filtering of multicast address, it will perfectly filter up to 64 multicast addresses if these addresses are chosen to map into unique locations in the multicast filter.



TL/F/10469-18

	D7	D6	D5	D4	D3	D2	D1	D0
MAR0	FB7	FB6	FB5	FB4	FB3	FB2	FB1	FB0
MAR1	FB15	FB14	FB13	FB12	FB11	FB10	FB9	FB8
MAR2	FB23	FB22	FB21	FB20	FB19	FB18	FB17	FB16
MAR3	FB31	FB30	FB29	FB28	FB27	FB26	FB25	FB24
MAR4	FB39	FB38	FB37	FB36	FB35	FB34	FB33	FB32
MAR5	FB47	FB46	FB45	FB44	FB43	FB42	FB41	FB40
MAR6	FB55	FB54	FB53	FB52	FB51	FB50	FB49	FB48
MAR7	FB63	FB62	FB61	FB60	FB59	FB58	FB57	FB56

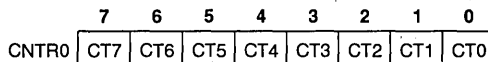
If address Y is found to hash to the value 32 (20H), then FB32 in MAR4 should be initialized to "1". This will cause the SNIC to accept any multicast packet with the address Y.

## 10.10 NETWORK TALLY COUNTERS

Three 8-bit counters are provided for monitoring the number of CRC errors, Frame Alignment Errors and Missed Packets. The maximum count reached by any counter is 192 (COH). These registers will be cleared when read by the CPU. The count is recorded in binary in CT0–CT7 of each Tally Register.

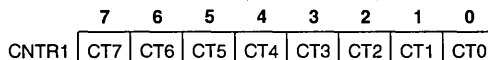
### Frame Alignment Error Tally (CNTR0)

This counter increments every time a packet is received with a Frame Alignment Error. The packet must have been recognized by the address recognition logic. The counter is cleared after it is read by the processor.



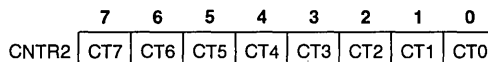
### CRC Error Tally (CNTR1)

This counter is incremented every time a packet is received with a CRC error. The packet must first be recognized by the address recognition logic. The counter is cleared after it is read by the processor.



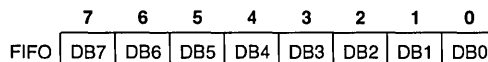
### Frames Lost Tally Register (CNTR2)

This counter is incremented if a packet cannot be received due to lack of buffer resources. In monitor mode, this counter will count the number of packets that pass the address recognition logic.



### FIFO

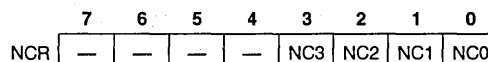
This is an 8-bit register that allows the CPU to examine the contents of the FIFO after loopback. The FIFO will contain the last 8 data bytes transmitted in the loopback packet. Sequential reads from the FIFO will advance a pointer in the FIFO and allow reading of all 8 bytes.



**Note:** The FIFO should only be read when the SNIC has been programmed in loopback mode.

### NUMBER OF COLLISIONS (NCR)

This register contains the number of collisions a node experiences when attempting to transmit a packet. If no collisions are experienced during a transmission attempt, the COL bit of the TSR will not be set and the contents of NCR will be zero. If there are excessive collisions, the ABT bit in the TSR will be set and the contents of NCR will be zero. The NCR is cleared after the TXP bit in the CR is set.



## 11.0 Initialization Procedures

The SNIC must be initialized prior to transmission or reception of packets from the network. Power on reset is applied to the SNIC's reset pin. This clears/sets the following bits:

Register	Reset Bits	Set Bits
Command Register (CR)	TXP, STA	RD2, STP
Interrupt Status (ISR)		RST
Interrupt Mask (IMR)	All Bits	
Data Control (DCR)		LAS
Transmit Config. (TCR)	LB1, LB0	

The SNIC remains in its reset state until a Start Command is issued. This guarantees that no packets are transmitted or received and that the SNIC remains a bus slave until all appropriate internal registers have been programmed. After initialization the STP bit of the command register is reset and packets may be received and transmitted.

### Initialization Sequence

The following initialization procedure is mandatory.

1. Program Command Register for Page 0 (Command Register = 21H)
2. Initialize Data Configuration Register (DCR)
3. Clear Remote Byte Count Registers (RBCR0, RBCR1)
4. Initialize Receive Configuration Register (RCR)
5. Place the SNIC in LOOPBACK mode 1 or 2 (Transmit Configuration Register = 02H or 04H)
6. Initialize Receive Buffer Ring: Boundary Pointer (BNDRY), Page Start (PSTART), and Page Stop (PSTOP)
7. Clear Interrupt Status Register (ISR) by writing 0FFH to it.
8. Initialize Interrupt Mask Register (IMR)
9. Program Command Register for page 1 (Command Register = 61H)
  - I) Initialize Physical Address Registers (PAR0-PAR5)
  - II) Initialize Multicast Address Registers (MAR0-MAR5)
  - III) Initialize CURRent pointer
10. Put SNIC in START mode (Command Register = 22H).
11. Initialize the Transmit Configuration for the intended value. The SNIC is now ready for transmission and reception.

Before receiving packets, the user must specify the location of the Receive Buffer Ring. This is programmed in the Page Start and Page Stop Registers. In addition, the Boundary and Current Page Register must be initialized to the value of the Page Start Register. These registers will be modified during reception of packets.

## 12.0 Loopback Diagnostics

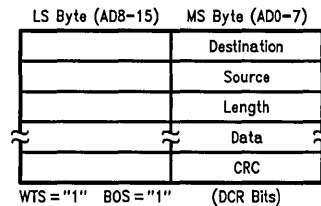
Three forms of local loopback are provided on the SNIC. The user has the ability to loopback through the deserializer on the controller, through the ENDEC module or the Coax Transceiver on the DP83901A SNIC. **Because of the half duplex architecture of the SNIC, loopback testing is a special mode of operation with the following restrictions:**

### Restrictions During Loopback

The FIFO is split into two halves, one used for transmission the other for reception. Only 8-bit fields can be fetched from memory so two tests are required for 16-bit systems to verify integrity of the entire data path. During loopback the maximum latency from the assertion of BREQ to BACK is 2.0  $\mu$ s. Systems that wish to use the loopback test yet do not meet this latency can limit the loopback to 7 bytes without experiencing underflow. Only the last 8 bytes of the loopback packet are retained in the FIFO. The last 8 bytes can be read through the FIFO register which will advance through the FIFO to allow reading the receive packet sequentially.

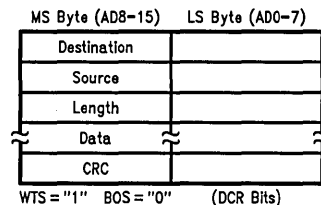
Destination Address	= (6 Bytes) Station Physical Address
Source Address	
Length	2 Bytes
Data	= 46 to 1500 Bytes
CRC	Appended by SNIC if CRC = "0" in TCR

When in word-wide mode with Byte Order Select set, the loopback packet must be assembled in the even byte location as shown below. (The loopback only operated with byte wide transfers.)



TL/F/10469-50

When in word-wide mode with Byte Order Select low, the following format must be used for the loopback packet.



TL/F/10469-51

**Note:** When using loopback in word mode 2n bytes must be programmed in TBCR0, 1. Where n = actual number of bytes assembled in even or odd location.

## 12.0 Loopback Diagnostics (Continued)

To initiate a loopback the user first assembles the loopback packet then selects the type of loopback using the Transmit Configuration register bits LB0, LB1. The transmit configuration register must also be set to enable or disable CRC generation during transmission. The user then issues a normal transmit command to send the packet. During loopback the receiver checks for an address match and if CRC bit in the TCR is set, the receiver will also check the CRC. The last 8 bytes of the loopback packet are buffered and can read out of the FIFO using FIFO read port.

### Loopback Modes

**MODE 1:** Loopback through the NIC Module (LB1 = 0, LB0 = 1): If this loopback is used, the NIC Modules's serial-izer is connected to the deserial-izer.

**MODE 2:** Loopback through the ENDEC Module (LB1 = 1, LB0 = 0): If the loopback is to be performed through the SNI, the SNIC provides a control (LPBK) that forces the ENDEC module to loopback all signals.

**MODE 3:** Loopback to Coax (LB1 = 1, LB0 = 1). Packets can be transmitted to the coax in loopback mode to check all of the transmit and receive paths and the coax itself.

**Note:** Collision and Carrier Sense can be generated by the ENDEC module and are masked by the NIC module. It is not possible to go directly between the loopback modes, it is necessary to return to normal operation (00H) when changing modes.

**Note:** The FIFO may only be read during Loopback. Reading the FIFO at any other time will cause the SNIC to malfunction.

### Reading the Loopback Packet

The last 8 bytes of a received packet can be examined by 8 consecutive reads of the FIFO register. The FIFO pointer is incremented after the rising edge of the CPU's read strobe by internally synchronizing and advancing the pointer. This may take up to four bus clock cycles, if the pointer has not been incremented by the time the CPU reads the FIFO register again, the SNIC will insert wait states.

**Note:** The FIFO may only be read during Loopback. Reading the FIFO at any other time will cause the SNIC to malfunction.

### Alignment of the Received Packet in the FIFO

Reception of the packet in the FIFO begins at location zero, after the FIFO pointer reaches the last location in the FIFO, the pointer wraps to the top of the FIFO overwriting the previously received data. This process continued until the last byte is received. The SNIC then appends the received byte count in the next two locations of the FIFO. The contents of the Upper Byte Count are also copied to the next FIFO location. The number of bytes used in the loopback packet determined the alignment of the packet in the FIFO.

The alignment for a 64-byte packet is shown below.

FIFO Location	FIFO Contents	
0	Lower Byte Count	First Byte Read
1	Upper Byte Count	Second Byte Read
2	Upper Byte Count	•
3	Last Byte	•
4	CRC1	•
5	CRC2	•
6	CRC3	•
7	CRC4	Last Byte Read

For the following alignment in the FIFO the packet length should be  $(N \times 8) + 5$  Bytes. Note that if the CRC bit in the TCR is set, CRC will not be appended by the transmitter. If the CRC is appended by the transmitter, the 1st four bytes, bytes N-3 to N, correspond to the CRC.

FIFO Location	FIFO Contents	
0	Byte N-4	First Byte Read
1	Byte N-3 (CRC1)	Second Byte Read
2	Byte N-2 (CRC2)	•
3	Byte N-1 (CRC3)	•
4	Byte N (CRC4)	•
5	Lower Byte Count	•
6	Upper Byte Count	Last Byte Read
7	Upper Byte Count	

### LOOPBACK TESTS

Loopback capabilities are provided to allow certain tests to be performed to validate operation of the DP83901A SNIC prior to transmitting and receiving packets on a live network. Typically these tests may be performed during power up of a node. The diagnostic provides support to verify the following:

1. Verify integrity of data path. Received data is checked against transmitted data.
2. Verify CRC logic's capability to generate good CRC on transmit, verify CRC on receive (good or bad CRC).



## 12.0 Loopback Diagnostics (Continued)

3. Verify that the Address Recognition Logic can

- a) Recognize address match packets
- b) Reject packets that fail to match an address

### LOOPBACK OPERATION IN THE SNIC

Loopback is a modified form of transmission using only half of the FIFO. This places certain restrictions on the use of loopback testing. When loopback mode is selected in the TCR, the FIFO is split. A packet should be assembled in memory with programming of TPSR and TBCR0, TBCR1 registers. When the transmit command is issued the following operations occur:

#### Transmitter Actions

1. Data is transferred from memory by the DMA until the FIFO is filled. For each transfer TBCR0 and TBCR1 are decremented. (Subsequent burst transfers are initiated when the number of bytes in the FIFO drops below the programmed threshold.)
2. The SNIC generates 56 bits of preamble followed by an 8-bit synch pattern.
3. Data transferred from FIFO to serializer.
4. If CRC = 1 in TCR, no CRC calculated by SNIC, the last byte transmitted is the last byte from the FIFO (Allows software CRC to be appended). If CRC = 0, SNIC calculates and appends four bytes of CRC.
5. At end of Transmission PTX bit set in ISR.

#### Receiver Actions

1. Wait for synch, all preamble stripped.
2. Store packet in FIFO, increment receive byte count for each incoming byte.
3. If CRC = 1 in TRC, receiver checks incoming packet for CRC errors. If CRC = 0 in TCR, receiver does not check CRC errors, CRC error bit always set in RSR (for address matching packets).
4. At end of receive, receive byte count written into FIFO, receive status register is updated. The PRX bit is typically set in the RSR even if the address does not match. If CRC errors are forced, the packet must match the address filters in order for the CRC error bit in the RSR to be set.

#### EXAMPLES

The following examples show what results can be expected from a properly operating SNIC during loopback. The restrictions and results of each type of loopback are listed for reference. The loopback tests are divided into two sets of tests. One to verify the data path, CRC generation and byte count through all three paths. The second set of tests uses internal loopback to verify the receiver's CRC checking and address recognition. For all of the tests the DCR was programmed to 40H.

Path	TCR	RCR	TSR	RSR	ISR
SNIC Internal	02	1F	53 (Note 1)	02 (Note 2)	02 (Note 3)

**Note 1:** Since carrier sense and collision detect are generated in the EN-DEC module. They are blocked during NIC loopback, carrier and CD heartbeat are not seen and the CRS and CDH bits are set.

**Note 2:** CRC errors are always indicated by receiver if CRC is appended by the transmitter.

**Note 3:** Only the PTX bit in the ISR is set, the PRX bit is only set if status is written to memory. In loopback this action does not occur and the PRX bit remains 0 for all loopback modes.

**Note 4:** All values are hex.

Path	TCR	RCR	TSR	RSR	ISR
SNIC Internal	04	1F	43 (Note 1)	02	02

**Note 1:** CDH is set, CRS is not set since it is generated by the external encoder/decoder.

Path	TCR	RCR	TSR	RSR	ISR
SNIC External	06	1F	03 (Note 1)	02	02 (Note 2)

**Note 1:** CDH and CRS should not be set. The TSR however, could also contain 01H, 03H, 07H and a variety of other values depending on whether collisions were encountered or the packet was deferred.

**Note 2:** Will contain 08H if packet is not transmittable.

**Note 3:** During external loopback the SNIC is now exposed to network traffic, it is therefore possible for the contents of both the Receive portion of the FIFO and the RSR to be corrupted by any other packet on the network. Thus in a live network the contents of the FIFO and RSR should not be depended on. The SNIC will still abide by the standard CSMA/CD protocol in external loopback mode. (i.e., The network will not be disturbed by the loopback packet.)

**Note 4:** All values are hex.

### CRC AND ADDRESS RECOGNITION

The next three tests exercise the address recognition logic and CRC. These tests should be performed using internal loopback only so that the SNIC is isolated from interference from the network. These tests also require the capability to generate CRC in software.

The address recognition logic cannot be directly tested. The CRC and FAE bits in the RSR are only set if the address in the packet matches the address filters. If errors are expected to be set and they are not set, the packet has been rejected on the basis of an address mismatch. The following sequence of packets will test the address recognition logic. The DCR should be set to 40H, the TCR should be set to 03H with a software generated CRC.

Packet Contents			Results
Test	Address	CRC	RSR
Test A	Matching	Good	01 (Note 1)
Test B	Matching	Bad	02 (Note 2)
Test C	Non-Matching	Bad	01

**Note 1:** Status will read 21H if multicast address used.

**Note 2:** Status will read 22H if multicast address used.

**Note 3:** In test A, the RSR is set up. In test B the address is found to match since the CRC is flagged as bad. Test C proves that the address recognition logic can distinguish a bad address and does not notify the RSR of the bad CRC. The receiving CRC is proven to work in test A and test B.

**Note 4:** All values are hex.

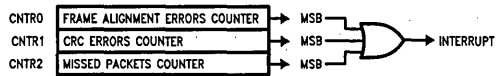
### NETWORK MANAGEMENT FUNCTIONS

Network management capabilities are required for maintenance and planning of a local area network. The SNIC supports the minimum requirement for network management in hardware, the remaining requirements can be met with software counts. There are three events that software alone can not track during reception of packets: CRC errors, Frame Alignment errors, and missed packets.

## 12.0 Loopback Diagnostics (Continued)

Since errored packets can be rejected, the status associated with these packets is lost unless the CPU can access the Receive Status Register before the next packet arrives. In situations where another packet arrives very quickly, the CPU may have no opportunity to do this. The SNIC counts the number of packets with CRC errors and Frame Alignment errors. 8-Bit counters have been selected to reduce overhead. The counters will generate interrupts whenever their MSBs are set so that a software routine can accumulate the network statistics and reset the counter before overflow occurs. The counters are sticky so that when they reach a count of 192 (C0H) counting is halted. An additional counter is provided to count the number of packets the SNIC misses due to buffer overflow or being offline.

The structure of the counters is shown below:



TL/F/10469-19

Additional information required for network management is available in the Receive and Transmit Status Registers. Transmit status is available after each transmission for information regarding events during transmission.

Typically, the following statistics might be gathered in software:

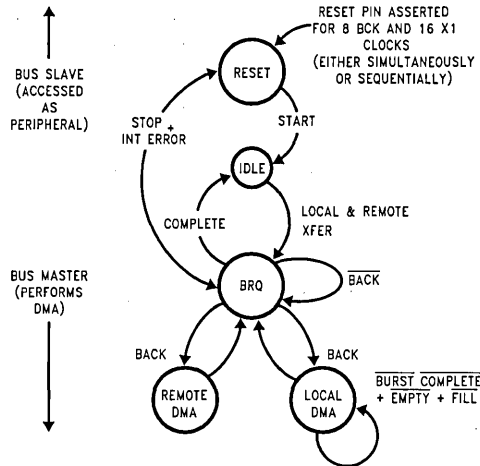
Traffic: Frames Sent OK  
 Frames Received OK  
 Multicast Frames Received  
 Packets Lost Due to Lack of Resources  
 Retries/Packet

Errors: CRC Errors  
 Alignment Errors  
 Excessive Collisions  
 Packet with Length Errors  
 Heartbeat Failure

### 13.0 Bus Arbitration and Timing

The SNIC operates in three possible modes:

- BUS MASTER (WHILE PERFORMING DMA)
- BUS SLAVE (WHILE BEING ACCESSED BY CPU)
- IDLE



TL/F/10469-20

Upon power-up the SNIC is in an indeterminate state. After receiving a hardware reset the SNIC is a bus slave in the Reset State, the receiver and transmitter are both disabled in this state. The reset state can be re-entered under three conditions, soft reset (Stop Command), hard reset (RESET input) or an error that shuts down the receiver of transmitter (FIFO underflow or overflow). After initialization of registers, the SNIC is issued a Start command and the SNIC enters Idle state. The idle state is exited by a request from the FIFO on the case of receiver or transmit, or from the Remote DMA in the case of Remote DMA operation. After acquiring

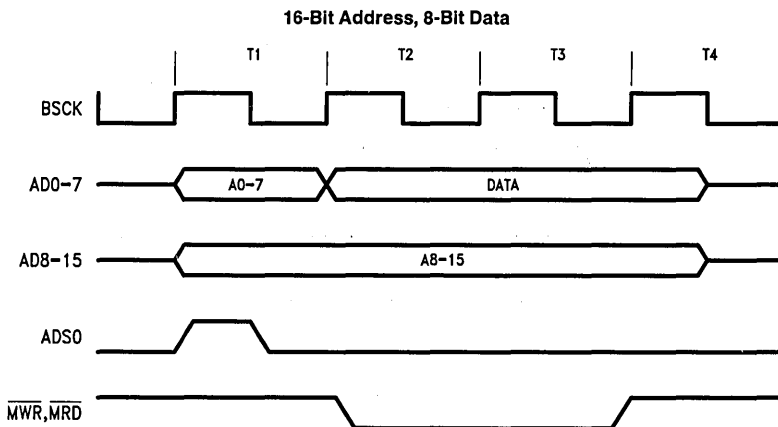
the bus in a BREQ/BACK handshake the Remote or Local DMA transfer is completed and the SNIC re-enters the idle state.

#### DMA TRANSFERS TIMING

The DMA can be programmed for the following types of transfers:

- 16-Bit Address, 8-bit Data Transfer
- 16-Bit Address, 16-bit Data Transfer
- 32-Bit Address, 8-bit Data Transfer
- 32-Bit Address, 16-bit Data Transfer

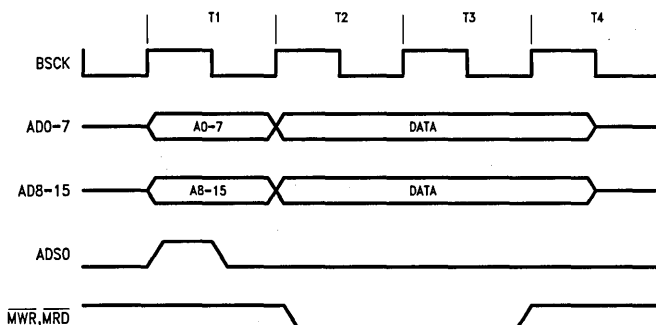
All DMA transfers use BSCK for timing. 16-Bit Address modes require 4 BSCK cycles as shown below:



TL/F/10469-21

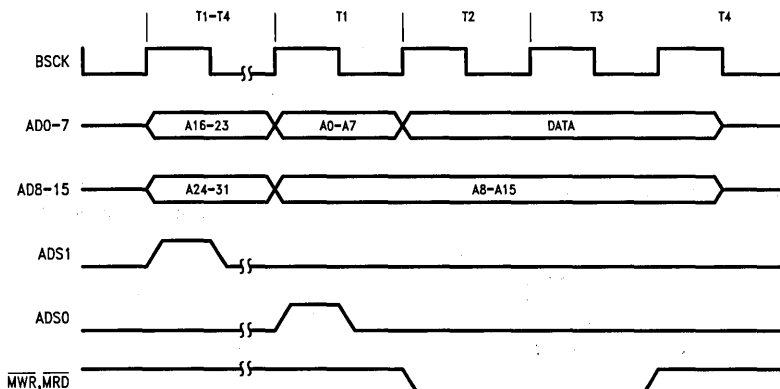
### 13.0 Bus Arbitration and Timing (Continued)

16-Bit Address, 16-Bit Data



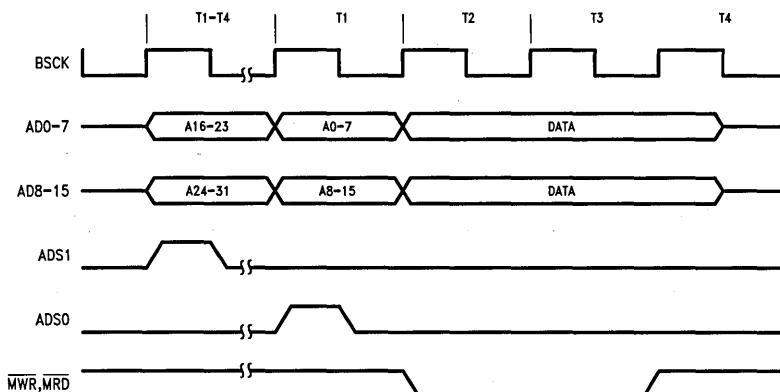
TL/F/10469-22

32-Bit Address, 8-Bit Data



TL/F/10469-23

32-Bit Address, 16-Bit Data



TL/F/10469-24

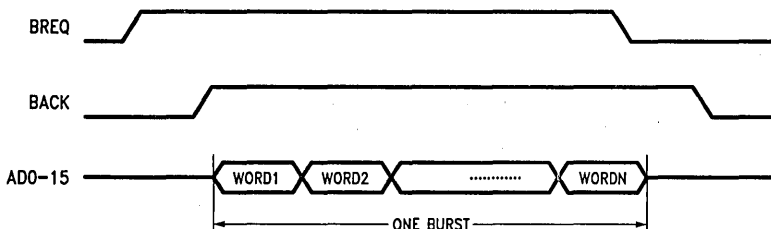
**Note:** In 32-bit address mode, ADS1 is at TRI-STATE after the first T1-T4 states; thus, a 4.7k pull-down resistor is required for 32-bit address.

### 13.0 Bus Arbitration and Timing (Continued)

When in 32-bit mode four additional BSCK cycles are required per burst. The first bus cycle (T1'-T4') of each burst is used to output the upper 16-bit addresses. This 16-bit address is programmed in RSAR0 and RSAR1 and points to a 64k page of system memory. All transmitted or received packets are constrained to reside within this 64k page.

#### FIFO BURST CONTROL

All Local DMA transfers are burst transfers, once the DMA requests the bus and the bus is acknowledged, the DMA will transfer an exact burst of bytes programmed in the Data Configuration Register (DCR) then relinquish the bus. If there are remaining bytes in the FIFO the next burst will not be initiated until the FIFO threshold is exceeded. If BACK is removed during the transfer, the burst transfer will be aborted. **(DROPPING BACK DURING A DMA CYCLE IS NOT RECOMMENDED.)**



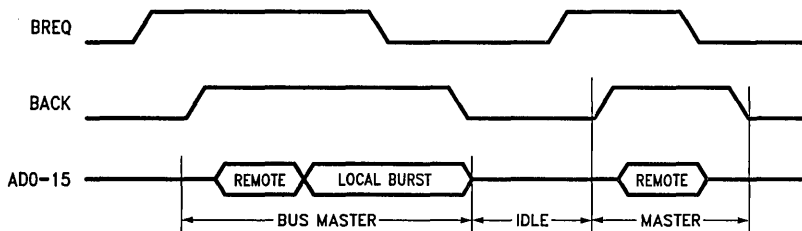
TL/F/10469-25

where N = 1, 2, 4, or 6 Words or N = 2, 4, 8, or 12 Bytes when in byte mode.

#### INTERLEAVED LOCAL OPERATION

If a remote DMA transfer is initiated or in progress when a packet is being received or transmitted, the Remote DMA transfer will be interrupted for higher priority Local DMA

transfers. When the Local DMA transfer is completed the Remote DMA will re-arbitrate for the bus and continue its transfers. This is illustrated below:



TL/F/10469-26

Note that if the FIFO requires service while a remote DMA is in progress, BREQ is not dropped and the Local DMA burst is appended to the Remote Transfer. When switching from a local transfer to a remote transfer, however, BREQ is dropped and raised again. This allows the CPU or other devices to fairly contend for the bus.

#### FIFO AND BUS OPERATIONS

##### Overview

To accommodate the different rates at which data comes from (or goes to) the network and goes to (or comes from) the system memory, the SNIC contains a 16-byte FIFO for buffering data between the bus and the media. The FIFO threshold is programmable, allowing filling (or emptying) the FIFO at different rates. When the FIFO has filled to its programmed threshold, the local DMA channel transfers these bytes (or words) into local memory. It is crucial that the local DMA is given access to the bus within a minimum bus latency time; otherwise a FIFO underrun (or overrun) occurs.

To understand FIFO underruns or overruns, there are two causes which produce this condition—

1. the bus latency is so long that the FIFO has filled (or emptied) from the network before the local DMA has serviced the FIFO.
2. the bus latency or bus data rate has slowed the throughput of the local DMA to a point where it is slower than the network data rate (10 Mb/s). This second condition is also dependent upon DMA clock and word width (byte wide or word wide).

The worst case condition ultimately limits the overall bus latency which the SNIC can tolerate.

##### FIFO Underrun and Transmit Enable

During transmission, if a FIFO underrun occurs, the Transmit enable (TXE) output may remain high (active). Generally, this will cause a very large packet to be transmitted onto the network. The jabber feature of the transceiver will terminate the transmission, and reset TXE.

To prevent this problem, a properly designed system will not allow FIFO underruns by giving the SNIC a bus acknowledge within time shown in the maximum bus latency curves shown and described later.

### 13.0 Bus Arbitration and Timing (Continued)

#### FIFO AT THE BEGINNING OF RECEIVE

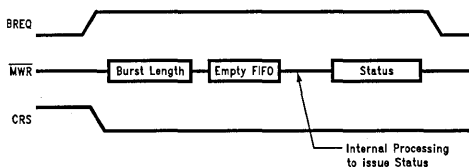
At the beginning of reception, the SNIC stores entire Address field of each incoming packet in the FIFO to determine whether the packet matches its Physical Address Register or maps to one of its Multicast Registers. This causes the FIFO to accumulate 8 bytes. Furthermore, there are some synchronization delays in the DMA PLA. Thus, the actual time that BREQ is asserted from the time the Start of Frame Delimiter (SFD) is detected is 7.8  $\mu\text{s}$ . This operation affects the bus latencies at 2 byte and 4 byte thresholds during the first receive BREQ since the FIFO must be filled to 8 bytes (or 4 words) before issuing a BREQ.

#### FIFO Operation at the End of Receive

When Carrier Sense goes low, the SNIC enters its end of packet processing sequence, emptying its FIFO and writing the status information at the beginning of the packet, *Figure 5*. The SNIC holds onto the bus for the entire sequence. The longest time BREQ may be extended occurs when a packet ends just as the SNIC performs its last FIFO burst. The SNIC, in this case, performs a programmed burst transfer followed by flushing the remaining bytes in the FIFO, and completes by writing the header information to memory. The following steps occur during this sequence.

1. SNIC issues BREQ because the FIFO threshold has been reached
2. During the burst, packet ends, resulting in BREQ extended.
3. SNIC flushes remaining bytes from FIFO
4. SNIC performs internal processing to prepare for writing the header.
5. SNIC writes 4-byte (2-word) header
6. SNIC deasserts BREQ

End of Packet Processing



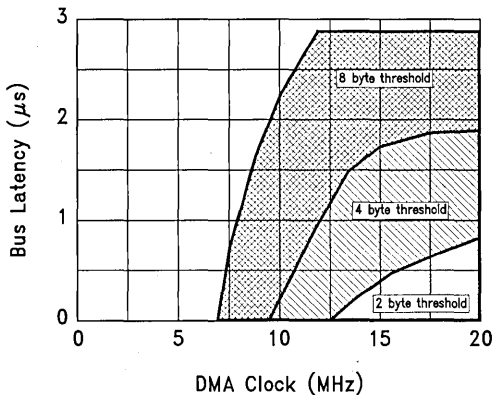
TL/F/10469-53

End of Packet Processing (EOPP) times for 10 MHz and 20 MHz have been tabulated below.

End of Packet Processing Times for Various FIFO Thresholds, Bus Clocks and Transfer Modes

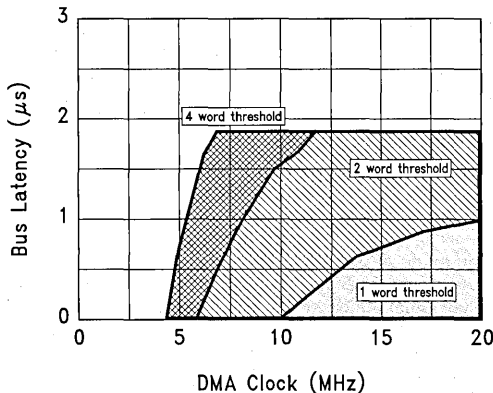
Mode	Threshold	Bus Clock	EOPP
Byte	2 bytes	10 MHz	7.0 $\mu\text{s}$
	4 bytes		8.6 $\mu\text{s}$
	8 bytes		11.0 $\mu\text{s}$
Byte	2 bytes	20 MHz	3.6 $\mu\text{s}$
	4 bytes		4.2 $\mu\text{s}$
	8 bytes		5.0 $\mu\text{s}$
Word	2 bytes	10 MHz	5.4 $\mu\text{s}$
	4 bytes		6.2 $\mu\text{s}$
	8 bytes		7.4 $\mu\text{s}$
Word	2 bytes	20 MHz	3.0 $\mu\text{s}$
	4 bytes		3.2 $\mu\text{s}$
	8 bytes		3.6 $\mu\text{s}$

Maximum Bus Latency for Byte Mode



TL/F/10469-54

Maximum Bus Latency for Word Mode



TL/F/10469-55

### 13.0 Bus Arbitration and Timing (Continued)

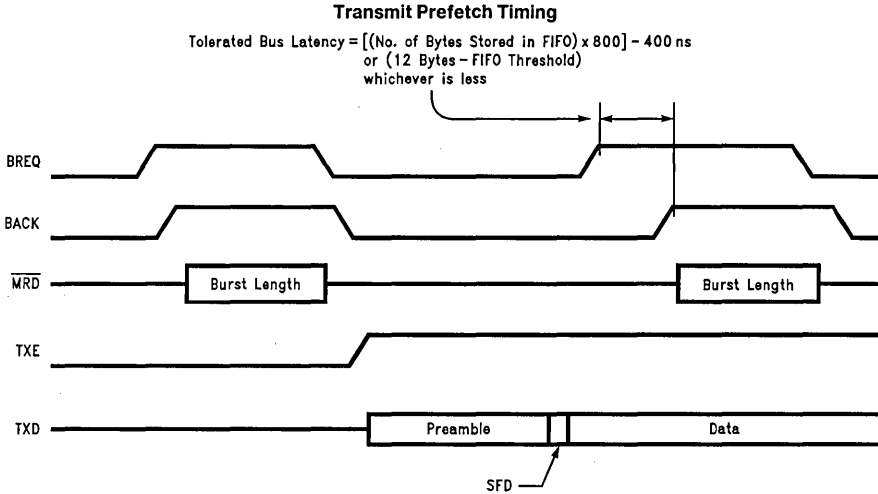
#### Threshold Detection (Bus Latency)

To assure that no overwriting of data in the FIFO occurs, the FIFO logic flags a FIFO overrun as the 13th byte is written into the FIFO, effectively shortening the FIFO to 13 bytes. The FIFO logic also operates differently in Byte Mode and in Word Mode. In Byte Mode, a threshold is indicated when the n+1 byte has entered the FIFO; thus, with an 8 byte threshold, the SNIC issues Bus Request (BREQ) when the 9th byte has entered the FIFO. For Word Mode, BREQ is not generated until the n+2 bytes have entered the FIFO. Thus, with a 4 word threshold (equivalent to 8 byte threshold), BREQ is issued when the 10th byte has entered the FIFO. The two graphs indicate the maximum allowable bus latency for Word or Byte transfer modes.

hold), BREQ is issued when the 10th byte has entered the FIFO. The two graphs indicate the maximum allowable bus latency for Word or Byte transfer modes.

#### The FIFO at the Beginning of Transmit

Before transmitting, the SNIC performs a prefetch from memory to load the FIFO. The number of bytes prefetched is the programmed FIFO threshold. The next BREQ is not issued until after the SNIC actually begins transmitting data, i.e., after SFD. The Transmit Prefetch Timing diagram illustrates this process.



TL/F/10469-56

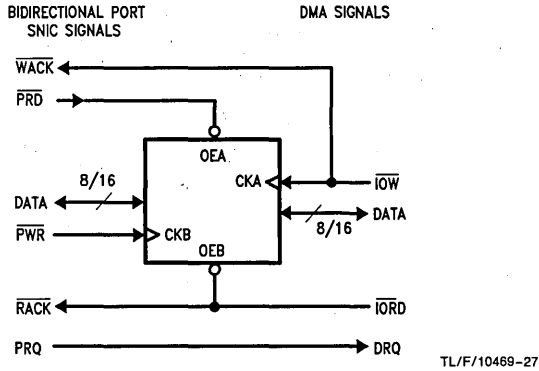
### 13.0 Bus Arbitration and Timing (Continued)

#### REMOTE DMA-BIDIRECTIONAL PORT CONTROL

The Remote DMA transfers data between the local buffer memory and a bidirectional port (memory to I/O transfer). This transfer is arbitrated on a byte by byte basis versus the burst transfer used for Local DMA transfers. This bidirec-

tional port is also read/written by the host. All transfers through this port are asynchronous. At any one time transfers are limited to one direction, either from the port to local buffer memory (Remote Write) or from local buffer memory to the port (Remote Read).

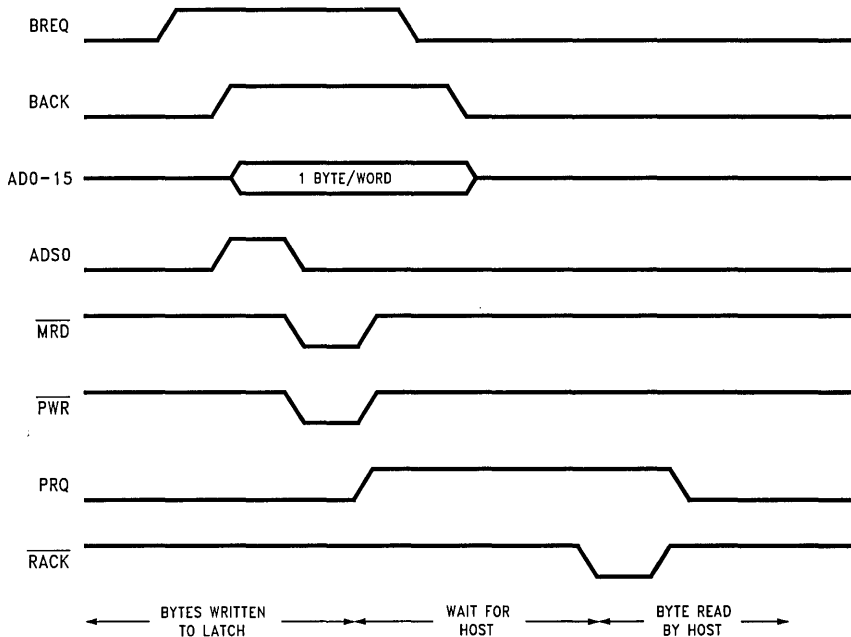
#### Bus Handshake Signals for Remote DMA Transfers



#### REMOTE READ TIMING

1. The DMA reads byte/word from local buffer memory and writes byte/word into latch, increments the DMA address and decrements the byte count (RBCR0, 1).
2. A Request Line (PRQ) is asserted to inform the system that a byte is available.
3. The system reads the port, the read strobe ( $\overline{RACK}$ ) is used as an acknowledge by the Remote DMA and it goes back to step 1.

Steps 1-3 are repeated until the remote DMA is complete. Note that in order for the Remote DMA to transfer a byte from memory to the latch, it must arbitrate access to the local bus via a BREQ, BACK handshake. After each byte or word is transferred to the latch, BREQ is dropped. If a Local DMA is in progress, the Remote DMA is held off until the local DMA is complete.





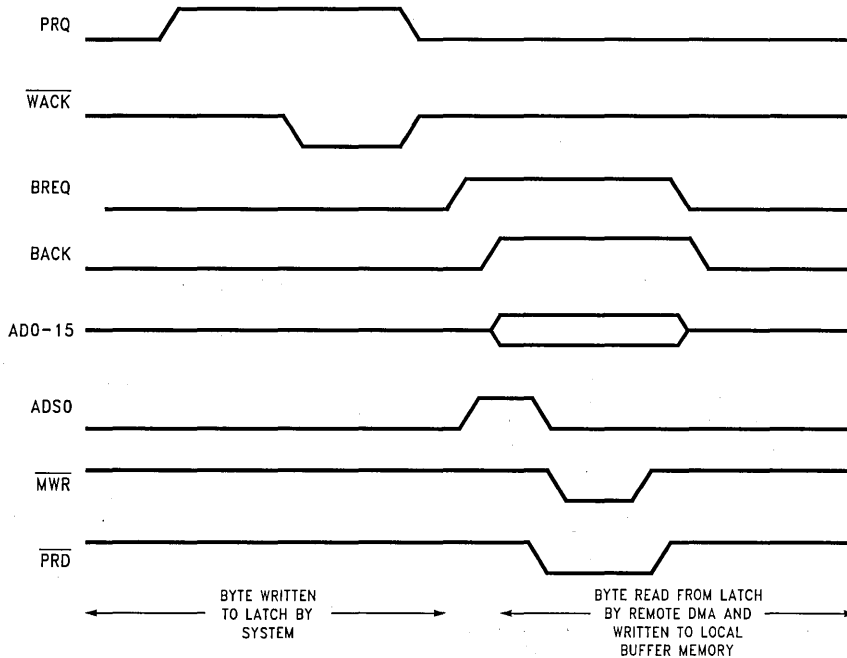
## 13.0 Bus Arbitration and Timing (Continued)

### REMOTE WRITE TIMING

A Remote Write operation transfers data from the I/O port to the local buffer RAM. The SNIC initiates a transfer by requesting a byte/word via the PRQ. The system transfers a byte/word to the latch via  $\overline{IOW}$ , this write strobe is detected by the SNIC and PRQ is removed. By removing the PRQ, the Remote DMA holds off further transfers into the latch until the current byte/word has been transferred from the latch, PRQ is reasserted and the next transfer can begin.

1. SNIC asserts PRQ. System writes byte/word into latch. SNIC removes PRQ.
2. Remote DMA reads contents of port and writes byte/word to local buffer memory, increments address and decrements byte count (RBCRO, 1).
3. Go back to step 1.

Steps 1–3 are repeated until the remote DMA is complete.



TL/F/10469-29

### REMOTE DMA WRITE

#### Setting PRQ Using the Remote Read

Under certain conditions the SNIC's bus state machine may issue  $\overline{MWR}$  and  $\overline{PRD}$  before PRQ for the first DMA transfer of a Remote Write Command. If this occurs this could cause data corruption, or cause the remote DMA count to be different from the main CPU count causing the system to "lock up."

To prevent this condition when implementing a Remote DMA Write, the Remote DMA Write command should first be preceded by a Remote DMA Read command to insure that the PRQ signal is asserted before the SNIC starts its port read cycle. The reason for this is that the state machine that asserts PRQ runs independently of the state machine that controls the DMA signals. The DMA machine assumes that PRQ is asserted, but actually may not be. To remedy this situation, a single Remote Read cycle should be inserted before the actual DMA Write Command is given. This will ensure that PRQ is asserted when the Remote DMA

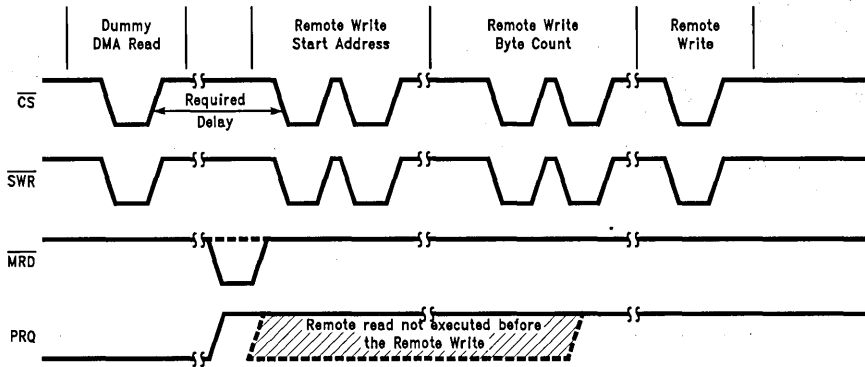
Write is subsequently executed. This single Remote Read cycle is called a "dummy Remote Read." In order for the dummy Remote Read cycle to operate correctly, the Start Address should be programmed to a known, safe location in the buffer memory space, and the Remote Byte Count should be programmed to a value greater than 1. This will ensure that the master read cycle is performed safely, eliminating the possibility of data corruption.

#### Remote Write with High Speed Buses

When implementing the Remote DMA Write solution with high speed buses and CPU's, timing problems may cause the system to hang. Therefore additional considerations are required.

The problem occurs when the system can execute the dummy Remote Read and then start the Remote Write before the SNIC has had a chance to execute the Remote Read. If this happens the PRQ signal will not get set, and the Remote Byte Count and Remote Start Address for the Remote

## 13.0 Bus Arbitration and Timing (Continued)



TL/F/10469-57

Note: The dashed lines indicate incorrect timing as described in the text.

**FIGURE 9. Timing Diagram for Dummy Remote Read**

Write operation could be corrupted. This is shown by the hatched waveforms in the timing diagram of *Figure 9*. The execution of the Remote Read can be delayed by the local DMA operations (particularly during end-of-packet processing).

To ensure the dummy Remote Read does execute, a delay must be inserted between writing the Remote Read Command, and starting to write the Remote Write State Address. (This time is designated in *Figure 9* by the delay arrows.) The recommended method to avoid this problem is after the Remote Read command is given, to poll both bytes of the Current Remote DMA Address Registers. When the address has incremented PRQ has been set. Software should recognize this and then start the Remote Write.

An additional caution for high speed systems is that the polling must follow guidelines specified in Time Between Chip Select section. That is, there must be at least 4 bus clocks between chip selects (for example when BSK = 20 MHz, then this time should be 200 ns).

The general flow for executing a Remote Write is:

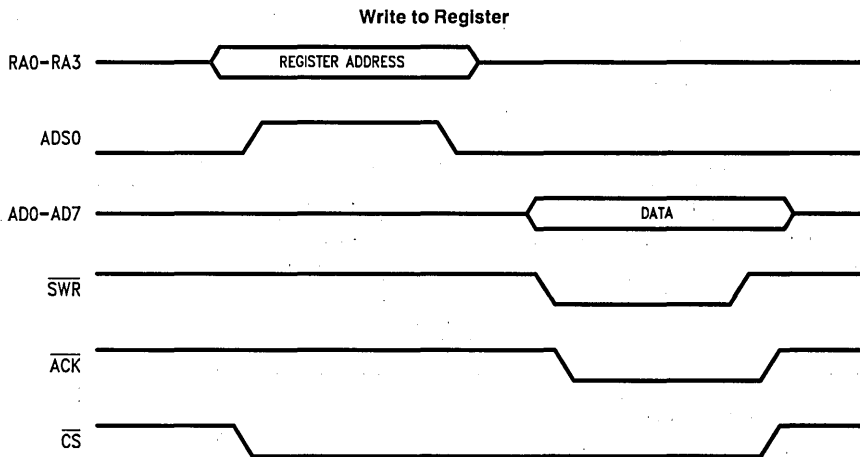
1. Set Remote Byte Count to a value  $> 1$  and Remote Start Address to unused RAM (one location before the transmit start address is usually a safe location).
2. Issue the "dummy" Remote Read command.
3. Read the Current Remote DMA Address (CRDA) (both bytes).
4. Compare to previous CRDA value if different go to 6.
5. Delay and jump to 3.
6. Set up for the Remote Write command, by setting the Remote Byte Count and the Remote Start Address (note that if Remote Byte count in step 1 can be set to the transmit byte count plus one, and the Remote Start Address to one less, these will now be incremented to the correct values.)
7. Issue the Remote Write command.

### 13.0 Bus Arbitration and Timing (Continued)

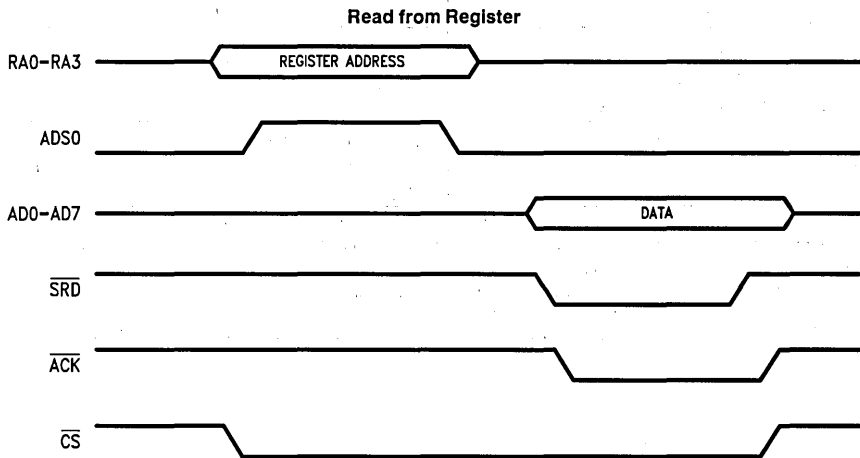
#### SLAVE MODE TIMING

When  $\overline{CS}$  is low, the SNIC becomes a bus slave. The CPU can then read or write any internal registers. All register access is byte wide. The timing for register access is shown below. The host CPU accesses internal registers with four address lines, RA0-RA3, SRD and SWR strobes.

ADS0 is used to latch the address when interfacing to a multiplexed, address data bus. Since the SNIC may be a local bus master when the host CPU attempts to read or write to the controller, an  $\overline{ACK}$  line is used to hold off the CPU until the SNIC leaves master mode. Some number of BSKC cycles is also required to allow the SNIC to synchronize to the read or write cycles.



TL/F/10469-30

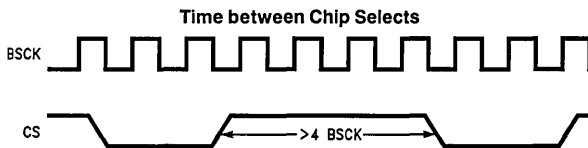


TL/F/10469-31

#### TIME BETWEEN CHIP SELECTS

The SNIC requires that successive chip selects be no closer than 4 bus clocks (BSKC) together. If the condition is violated, the SNIC may glitch  $\overline{ACK}$ . CPUs that operate from pipelined instructions (i.e., 386) or have a cache (i.e., 486) can

execute consecutive I/O cycles very quickly. The solution is to delay the execution of consecutive I/O cycles by either breaking the pipeline or forcing the CPU to access outside its cache.



TL/F/10469-58

## 14.0 Preliminary Electrical Characteristics

### Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage ( $V_{CC}$ )	-0.5V to +7.0V
DC Input Voltage ( $V_{IN}$ )	-0.5V to $V_{CC} + 0.5V$
DC Output Voltage ( $V_{OUT}$ )	-0.5V to $V_{CC} + 0.5V$
Storage Temperature Range ( $T_{STG}$ )	-65°C to +150°C
Power Dissipation (PD)	800 mW
Lead Temp. (TL) (Soldering, 10 sec.)	260°C
ESD Rating ( $R_{ZAP} = 1.5k, C_{ZAP} = 120 pF$ )	1.5 kV

Note: Absolute Maximum ratings are those values beyond which the safety of the device cannot be guaranteed. They are not meant to imply that the device should be operated at these limits.

Note: All specifications in this datasheet are valid only if the mandatory isolation is employed and all differential signals are taken to exist at the AUI side of the isolation.

### Preliminary DC Specifications $T_A = 0^\circ\text{C}$ to $70^\circ\text{C}$ , $V_{CC} = 5V \pm 5\%$ , unless otherwise specified.

Symbol	Parameter	Conditions	Min	Max	Units
$V_{OH}$	Minimum High Level Output Voltage (Notes 1, 4)	$I_{OH} = -20 \mu\text{A}$	$V_{CC} - 0.1$		V
		$I_{OH} = -2.0 \text{ mA}$	3.5		V
$V_{OL}$	Minimum Low Level Output Voltage (Notes 1, 4)	$I_{OL} = 20 \mu\text{A}$		0.1	V
		$I_{OL} = 2.0 \text{ mA}$		0.4	V
$V_{IH}$	Minimum High Level Input Voltage (Note 2)		2.0		V
$V_{IH2}$	Minimum High Level Input Voltage For RACK WACK (Note 2)		2.7		V
$V_{IL}$	Minimum Low Level Input Voltage (Note 2)			0.8	V
$V_{IL2}$	Minimum Low Level Input Voltage For RACK, WACK (Note 2)			0.6	V
$I_{IN}$	Input Current	$V_I = V_{CC}$ or GND	-1.0	+1.0	$\mu\text{A}$
$I_{OZ}$	Minimum TRI-STATE Output Leakage Current (Note 5)	$V_{OUT} = V_{CC}$ or GND	-10	+10	$\mu\text{A}$
$I_{CC}$	Average Supply Current (Note 3)	X1 = 20 MHz Clock $I_{OUT} = 0 \mu\text{A}$ $V_{IN} = V_{CC}$ or GND		110	mA

**Note 1:** These levels are tested dynamically using a limited amount of functional test patterns, please refer to AC test load.

**Note 2:** Limited functional test patterns are performed at these input levels. The majority of functional tests are performed at levels of 0V and 3V.

**Note 3:** This is measured with a 0.1  $\mu\text{F}$  bypass capacitor between  $V_{CC}$  and GND.

**Note 4:** The low drive CMOS compatible  $V_{OH}$  and  $V_{OL}$  limits are not tested directly. Detailed device characterization validates that this specification can be guaranteed by testing the high drive TTL compatible  $V_{OL}$  and  $V_{OH}$  specification.

**Note 5:** RA0-RA3, PRD, WACK, BREQ and INT pins are used as outputs in test mode and as a result are tested as if they are TRI-STATE input/outputs. For these pins the input leakage specification is  $I_{OZ}$ .

## 14.0 Preliminary Electrical Characteristics (Continued)

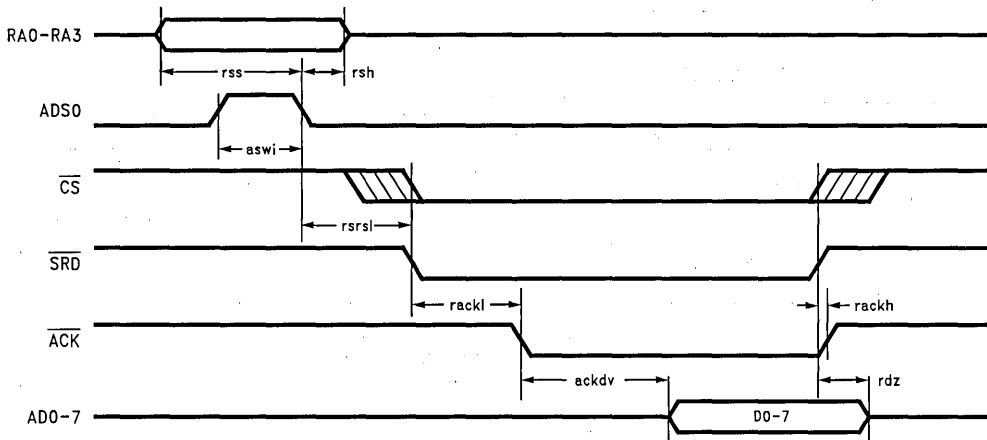
### Preliminary DC Specifications $T_A = 0^\circ\text{C}$ to $70^\circ\text{C}$ , $V = 5\text{V} \pm 5\%$ , unless otherwise specified.

Symbol	Parameter	Conditions	Min	Max	Units
<b>DIFFERENTIAL PINS (TX<math>\pm</math>, RX<math>\pm</math>, and CD<math>\pm</math>)</b>					
V <sub>OD</sub>	Diff. Output Voltage (TX $\pm$ )	78 $\Omega$ Termination, and 270 $\Omega$ s from each to GND	$\pm 550$	$\pm 1200$	mV
V <sub>OB</sub>	Diff. Output Voltage Imbalance (TX $\pm$ )	Same as Above	Typical: 40 mV		
V <sub>U</sub>	Undershoot Voltage (TX $\pm$ )	Same as Above	Typical: 80 mV		
V <sub>DS</sub>	Diff. Squelch Threshold (RX $\pm$ and CD $\pm$ )		-175	-300	mV
V <sub>CM</sub>	Diff. Input Common Mode Voltage (RX $\pm$ and CD $\pm$ ) (Note 1)		0	5.25	V
<b>OSCILLATOR PINS (X1 and GND/X2)</b>					
V <sub>IH</sub>	X1 Input High Voltage	X1 is Connected to an Oscillator and GND/X2 is Grounded	2.0		V
V <sub>IL</sub>	X1 Input Low Voltage	Same as Above		0.8	V
I <sub>OSC</sub>	X1 Input Current	GND/X2 is Grounded V <sub>IN</sub> = V <sub>CC</sub> or GND		+3	mA

Note 1: This parameter is guaranteed by the isolation and is not tested.

# 15.0 Switching Characteristics AC Specs DP83901A Note: All Timing is Preliminary

Register Read (Latched Using ADS0)



TL/F/10469-32

Symbol	Parameter	Min	Max	Units
rss	Register Select Setup to ADS0 Low	10		ns
rsh	Register Select Hold from ADS0 Low	13		ns
aswi	Address Strobe Width In	15		ns
ackdv	Acknowledge Low to Data Valid		55	ns
rdz	Read Strobe to Data TRI-STATE (Note 3)	15	70	ns
rackl	Read Strobe to $\overline{ACK}$ Low (Notes 1, 2)		$n \cdot \text{bcyc} + 30$	ns
rackh	Read Strobe to $\overline{ACK}$ High		30	ns
rsrsl	Register Select to Slave Read Low, Latched RS0-3	10		ns

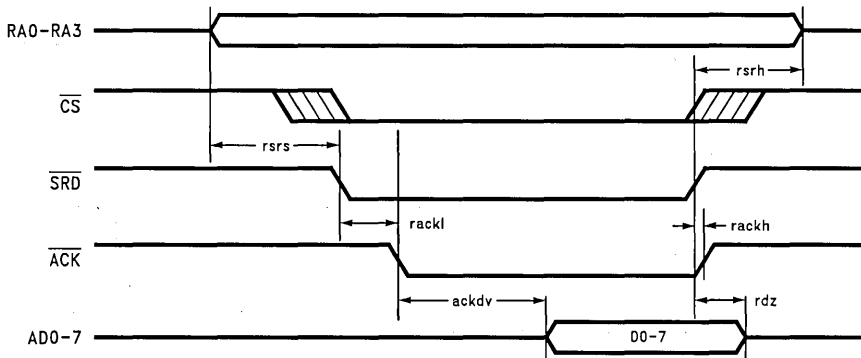
**Note 1:**  $\overline{ACK}$  is not generated until  $\overline{CS}$  and  $\overline{SRD}$  are low and the SNIC has synchronized to the register access. The SNIC will insert an integral number of Bus Clock cycles until it is synchronized. In Dual Bus systems additional cycles will be used for a local or remote DMA to complete. Wait states must be issued to the CPU until  $\overline{ACK}$  is asserted low.

**Note 2:**  $\overline{CS}$  may be asserted before or after  $\overline{SRD}$ . If  $\overline{CS}$  is asserted after  $\overline{SRD}$ , rackl is referenced from falling edge of  $\overline{CS}$ .  $\overline{CS}$  can be de-asserted concurrently with  $\overline{SRD}$  or after  $\overline{SRD}$  is de-asserted.

**Note 3:** These limits include the RC delay inherent in our test method. These signals typically turn off within 15 ns, enabling other devices to drive these lines with no contention.

### 15.0 Switching Characteristics AC Specs DP83901A Note: All Timing is Preliminary (Continued)

Register Read (Non-Latched, ADS0 = 1)



TL/F/10469-33

Symbol	Parameter	Min	Max	Units
rsrc	Register Select to Read Setup (Notes 1, 3)	10		ns
rsrh	Register Select Hold from Read	0		ns
ackdv	$\overline{\text{ACK}}$ Low to Valid Data		55	ns
rdz	Read Strobe to Data TRI-STATE (Note 2)	15	70	ns
rackl	Read Strobe to $\overline{\text{ACK}}$ Low (Note 3)		$n \cdot \text{bcyc} + 30$	ns
rackh	Read Strobe to $\overline{\text{ACK}}$ High		30	ns

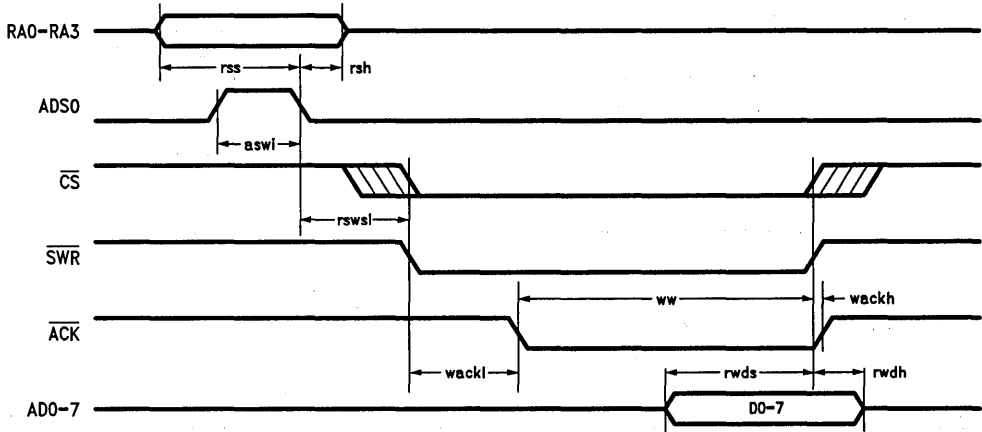
**Note 1:** rsrc includes flow-through time of latch.

**Note 2:** These limits include the RC delay inherent in our test method. These signals typically turn off within 15 ns enabling other devices to drive these lines with no contention.

**Note 3:**  $\overline{\text{CS}}$  may be asserted before or after RA0-3, and  $\overline{\text{SRD}}$ , since address decode begins when  $\overline{\text{ACK}}$  is asserted. If  $\overline{\text{CS}}$  is asserted after RA0-3, and  $\overline{\text{SRD}}$ , rackl is referenced from falling edge of  $\overline{\text{CS}}$ .

**15.0 Switching Characteristics** AC Specs DP83901A Note: All Timing is Preliminary (Continued)

**Register Write (Latched Using ADS0)**



TL/F/10469-34

Symbol	Parameter	Min	Max	Units
rss	Register Select Setup to ADS0 Low	10		ns
rsh	Register Select Hold from ADS0 Low	17		ns
aswi	Address Strobe Width In	15		ns
rwds	Register Write Data Setup	20		ns
rwdh	Register Write Data Hold	21		ns
ww	Write Strobe Width from $\overline{ACK}$	50		ns
wackh	Write Strobe High to $\overline{ACK}$ High		30	ns
wackl	Write Low to $\overline{ACK}$ Low (Notes 1, 2)		$n \cdot \text{bcyc} + 30$	ns
rswsl	Register Select to Write Strobe Low	10		ns

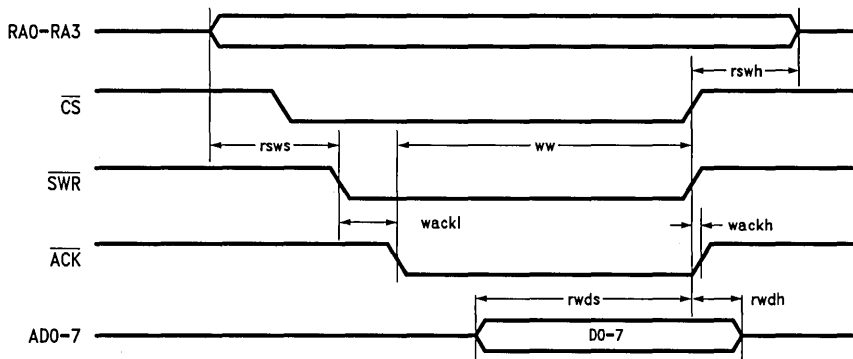
**Note 1:**  $\overline{ACK}$  is not generated until  $\overline{CS}$  and  $\overline{SWR}$  are low and the SNIC has synchronized to the register access. In Dual Bus Systems additional cycles will be used for a local DMA or Remote DMA to complete.

**Note 2:**  $\overline{CS}$  may be asserted before or after  $\overline{SWR}$ . If  $\overline{CS}$  is asserted after  $\overline{SWR}$ , wackl is referenced from falling edge of  $\overline{CS}$ .



**15.0 Switching Characteristics** AC Specs DP83901A **Note:** All Timing is Preliminary (Continued)

**Register Write (Non-Latched, ADS0 = 1)**



TL/F/10469-35

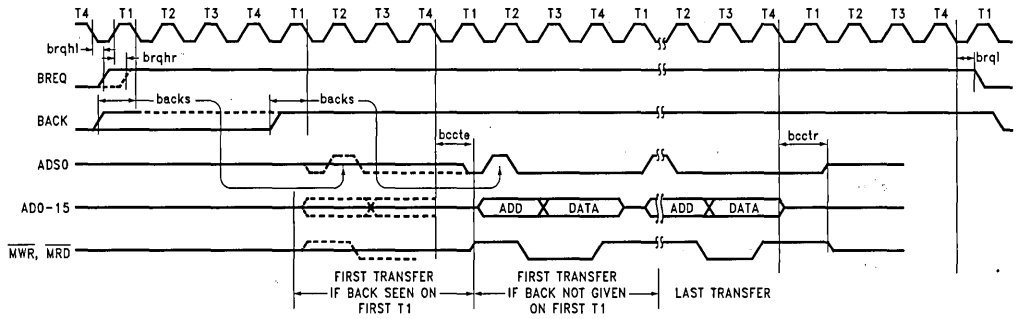
Symbol	Parameter	Min	Max	Units
rsws	Register Select to Write Setup (Note 1)	15		ns
rswh	Register Select Hold from Write	0		ns
rwds	Register Write Data Setup	20		ns
rwdh	Register Write Data Hold	21		ns
wackl	Write Low to $\overline{ACK}$ Low (Note 2)		$n \cdot bcyc + 30$	ns
wackh	Write High to $\overline{ACK}$ High		30	ns
ww	Write Width from $\overline{ACK}$	50		ns

**Note 1:** Assumes ADS0 is high when RA0-3 changing.

**Note 2:** ACK is not generated until CS and SWR are low and the SNIC has synchronized to the register access. In Dual Bus systems additional cycles will be used for a local DMA or remote DMA to complete.

# 15.0 Switching Characteristics AC Specs DP83901A **Note:** All Timing is Preliminary (Continued)

## DMA Control, Bus Arbitration



TL/F/10469-36

Symbol	Parameter	Min	Max	Units
brqhl	Bus Clock to Bus Request High for Local DMA		50	ns
brqhr	Bus Clock to Bus Request High for Remote DMA		45	ns
brql	Bus Request Low from Bus Clock		60	ns
backs	Acknowledge Setup to Bus Clock (Note 1)	2		ns
bccte	Bus Clock to Control Enable		60	ns
bcctr	Bus Clock to Control Release (Notes 2, 3)		70	ns

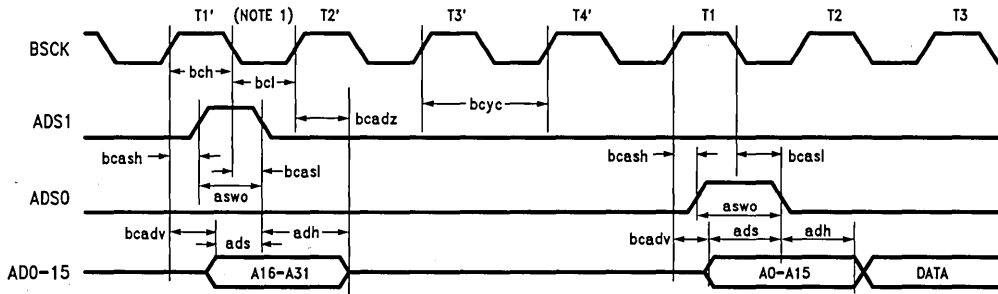
**Note 1:** BACK must be setup before T1 after BREQ is asserted. Missed setup will slip the beginning of the DMA by four bus clocks. The Bus Latency will influence the allowable FIFO threshold and transfer mode (empty/fill vs exact burst transfer).

**Note 2:** During remote DMA transfers only, a single bus transfer is performed. During local DMA operations burst mode transfers are performed.

**Note 3:** These limits include the RC delay inherent in our test method. These signals typically turn off within 15 ns enabling other devices to drive these lines with no contention.

## 15.0 Switching Characteristics AC Specs DP83901A Note: All Timing is Preliminary (Continued)

**DMA Address Generation**



TL/F/10469-37

Symbol	Parameter	Min	Max	Units
bcyc	Bus Clock Cycle Time (Note 2)	50	125	ns
bch	Bus Clock High Time	20		ns
bcl	Bus Clock Low Time	20		ns
bcash	Bus Clock to Address Strobe High		34	ns
bcal	Bus Clock to Address Strobe Low		44	ns
aswo	Address Strobe Width Out	bch		ns
bcadv	Bus Clock to Address Valid		45	ns
bcadz	Bus Clock to Address TRI-STATE (Note 3)	15	55	ns
ads	Address Setup to ADS0/1 Low	bch - 15		ns
adh	Address Hold from ADS0/1 Low	bcl - 5		ns

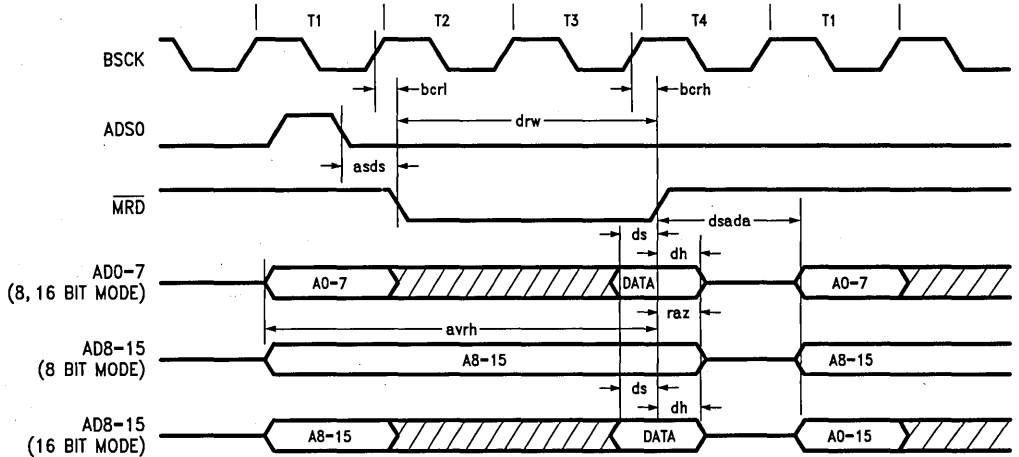
**Note 1:** Cycles T1', T2', T3' and T4' are only issued for the first transfer in a burst when 32-bit mode has been selected.

**Note 2:** The rate of bus clock must be high enough to support transfers to/from the FIFO at a rate greater than the serial network transfers from/to the FIFO.

**Note 3:** These limits include the RC delay inherent in our test method. These signals typically turn off within 15 ns, enabling other devices to drive these lines with no contention.

**15.0 Switching Characteristics** AC Specs DP83901A Note: All Timing is Preliminary (Continued)

**DMA Memory Read**



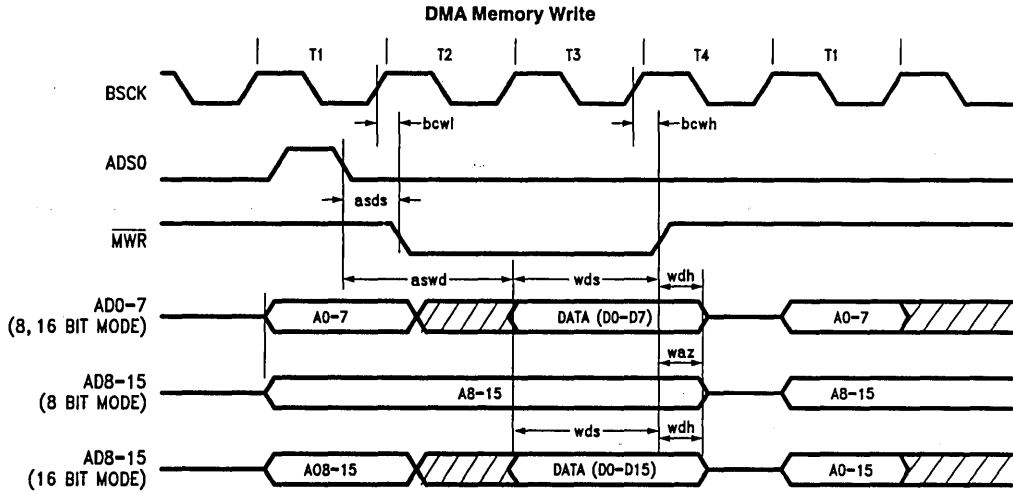
TL/F/10469-38

Symbol	Parameter	Min	Max	Units
bcr1	Bus Clock to Read Strobe Low		43	ns
bcrh	Bus Clock to Read Strobe High		40	ns
ds	Data Setup to Read Strobe High	22		ns
dh	Data Hold from Read Strobe High	0		ns
drw	DMA Read Strobe Width Out	$2 \cdot \text{bcyc} - 15$		ns
raz	Memory Read High to Address TRI-STATE (Notes 1, 2)		$\text{bch} + 40$	ns
asds	Address Strobe to Data Strobe		$\text{bcl} + 10$	ns
dsada	Data Strobe to Address Active	$\text{bcyc} - 10$		ns
avrh	Address Valid to Read Strobe High	$3 \cdot \text{bcyc} - 18$		ns

**Note 1:** During a burst A8-A15 are not TRI-STATE if byte wide transfers are selected. On the last transfer A8-A15 are TRI-STATE as shown above.

**Note 2:** These limits include the RC delay inherent in our test method. These signals typically turn off within  $\text{bch} + 15$  ns, enabling other devices to drive these lines with no contention.

**15.0 Switching Characteristics** AC Specs DP83901A **Note:** All Timing is Preliminary (Continued)



TL/F/10469-39

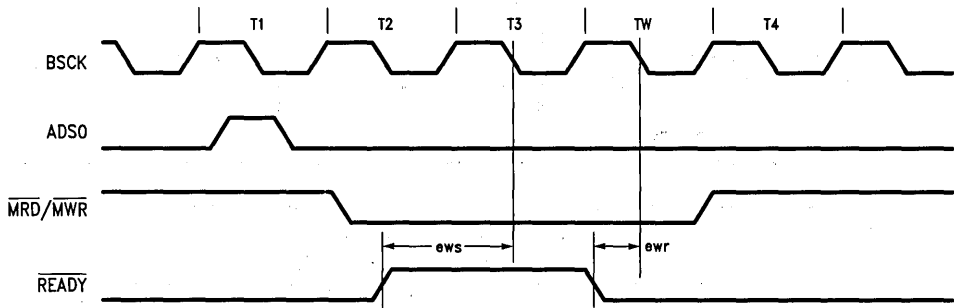
Symbol	Parameter	Min	Max	Units
bcwl	Bus Clock to Write Strobe Low		40	ns
bcwh	Bus Clock to Write Strobe High		40	ns
wds	Data Setup to $\overline{WR}$ High	$2 \cdot \text{bcyc} - 30$		ns
wdh	Data Hold from $\overline{WR}$ Low	$\text{bch} + 7$		ns
waz	Write Strobe to Address TRI-STATE (Notes 1, 2)		$\text{bch} + 40$	ns
asds	Address Strobe to Data Strobe		$\text{bcl} + 10$	ns
aswd	Address Strobe to Write Data Valid		$\text{bcl} + 30$	ns

**Note 1:** When using byte mode transfers A8-A15 are only TRI-STATE on the last transfer, waz timing is only valid for last transfer in a burst.

**Note 2:** These limits include the RC delay inherent in our test method. These signals typically turn off within  $\text{bch} + 15$  ns, enabling other devices to drive these lines with no contention.

# 15.0 Switching Characteristics AC Specs DP83901A Note: All Timing is Preliminary (Continued)

Wait State Insertion



TL/F/10469-40

Symbol	Parameter	Min	Max	Units
ews	External Wait Setup to T3 0Clock (Note 1)	10		ns
ewr	External Wait Release Time (Note 1)	15		ns

**Note 1:** The addition of wait states affects the count of deserialized bytes and is limited to a number of bus clock cycles depending on the bus clock and network rates. The allowable wait states are found in the table below. (Assumes 10 Mbit/sec data rate.)

BSCCK (MHz)	Max # of Wait States	
	Byte Transfer	Word Transfer
8	0	1
10	0	1
12	1	2
14	1	2
16	1	3
18	2	3
20	2	4

Table assumes 10 MHz network clock.

The number of allowable wait states in byte mode can be calculated using:

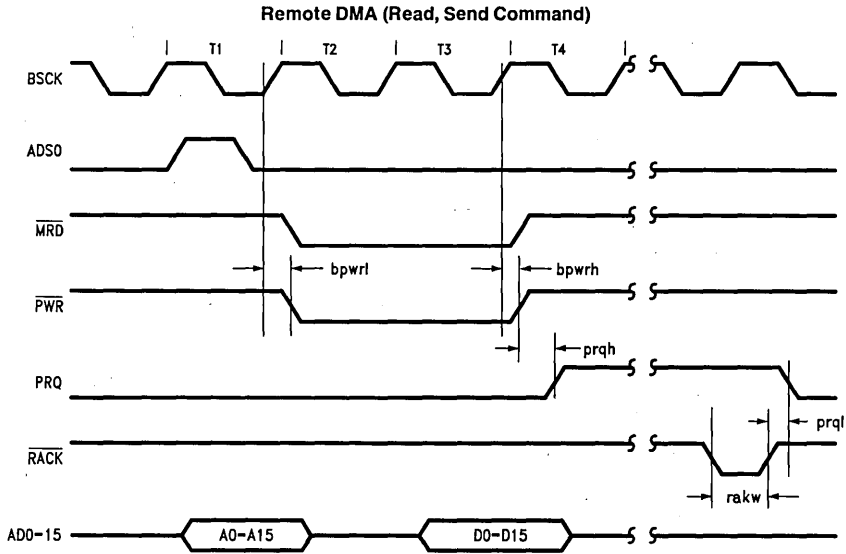
$$\#W_{(\text{byte mode})} = \left( \frac{8 \text{ tnw}}{4.5 \text{ tbsck}} - 1 \right)$$

- #W = Number of Wait States
- tnw = Network Clock Period
- tbsck = BSCCK Period

The number of allowable wait states in word mode can be calculated using:

$$\#W_{(\text{word mode})} = \left( \frac{5 \text{ tnw}}{2 \text{ tbsck}} - 1 \right)$$

**15.0 Switching Characteristics** AC Specs DP83901A **Note:** All Timing is Preliminary (Continued)



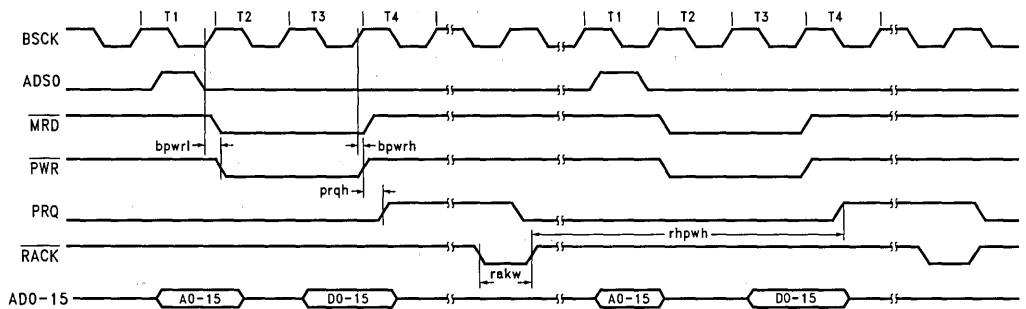
TL/F/10469-41

Symbol	Parameter	Min	Max	Units
bpwrl	Bus Clock to Port Write Low		43	ns
bpwrh	Bus Clock to Port Write High		40	ns
prqh	Port Write High to Port Request High (Note 1)		30	ns
prql	Port Request Low from Read Acknowledge High		60	ns
rakw	Remote Acknowledge Read Strobe Pulse Width	20		ns

**Note 1:** Start of next transfer is dependent on where RACK is generated relative to BSCK and whether a local DMA is pending.

# 15.0 Switching Characteristics AC Specs DP83901A Note: All Timing is Preliminary (Continued)

**Remote DMA (Read, Send Command) Recovery Time**



TL/F/10469-42

Symbol	Parameter	Min	Max	Units
bpwrl	Bus Clock to Port Write Low		43	ns
bpwrh	Bus Clock to Port Write High		40	ns
prqh	Port Write High to Port Request High (Note 1)		30	ns
prql	Port Request Low from Read Acknowledge High		60	ns
rakw	Remote Acknowledge Read Strobe Pulse Width	20		ns
rhpwh	Read Acknowledge High to Next Port Write Cycle (Notes 2, 3, 4)	11		BSCCK

**Note 1:** Start of next transfer is dependent on where RACK is generated relative to BSCCK and whether a local DMA is pending.

**Note 2:** This is not a measured value but guaranteed by design.

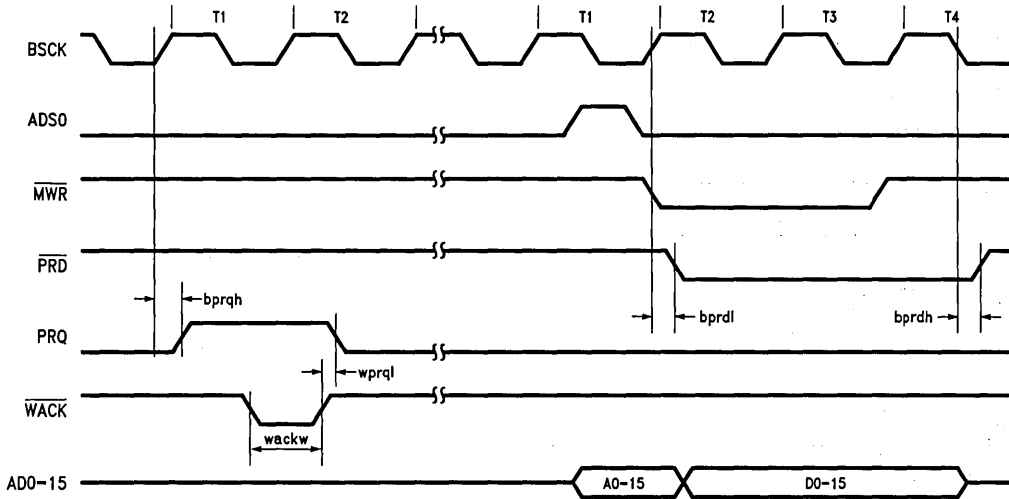
**Note 3:** RACK must be high for a minimum of 7 BSCCK.

**Note 4:** Assumes no local DMA interleave, no CS, and immediate BACK.



### 15.0 Switching Characteristics AC Specs DP83901A Note: All Timing is Preliminary (Continued)

Remote DMA (Write Cycle)



TL/F/10469-43

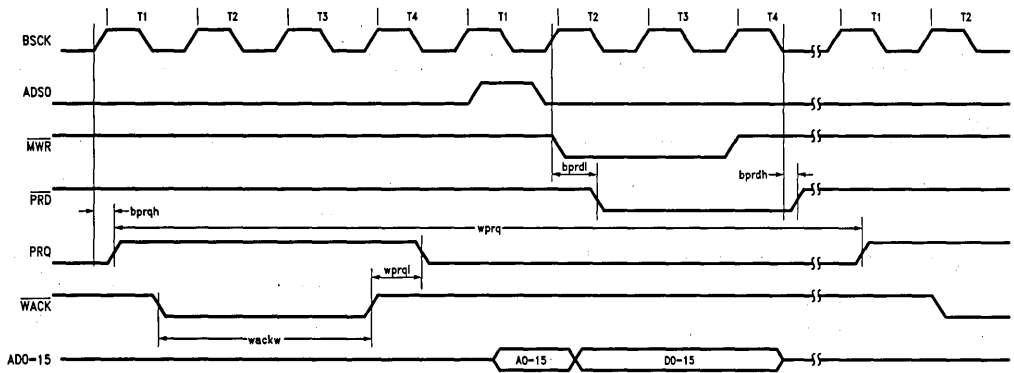
Symbol	Parameter	Min	Max	Units
bprqh	Bus Clock to Port Request High (Note 1)		42	ns
wprql	WACK to Port Request Low		52	ns
wackw	WACK Pulse Width	25		ns
bprdl	Bus Clock to Port Read Low (Note 2)		55	ns
bprdh	Bus Clock to Port Read High		40	ns

**Note 1:** The first port request is issued in response to the remote write command. It is subsequently issued on T1 clock cycles following completion of remote DMA cycles.

**Note 2:** The start of the remote DMA write following WACK is dependent on where WACK is issued relative to BSCCK and whether a local DMA is pending.

# 15.0 Switching Characteristics AC Specs DP83901A Note: All Timing is Preliminary (Continued)

## Remote DMA (Write Cycle) Recovery Time



TL/F/10469-44

Symbol	Parameter	Min	Max	Units
bprqh	Bus Clock to Port Request High (Note 1)		42	ns
wprql	WACK to Port Request Low		50	ns
wackw	WACK Pulse Width	25		ns
bprdl	Bus Clock to Port Read Low (Note 2)		55	ns
bprdh	Bus Clock to Port Read High		40	ns
wprq	Remote Write Port Request to Port Request Time (Notes 3, 4, 5)	12		BSCK

**Note 1:** The first port request is issued in response to the remote write command. It is subsequently issued on T1 clock cycles following completion of remote DMA cycles.

**Note 2:** The start of the remote DMA write following WACK is dependent on where WACK is issued relative to BSCK and whether a local DMA is pending.

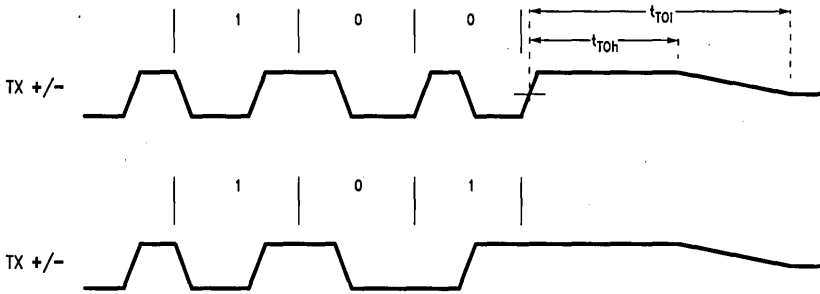
**Note 3:** Assuming wackw < 1 BSCK, and no local DMA interleave, no CS, immediate BACK, and WACK goes high before T4.

**Note 4:** WACK must be high for a minimum of 7 BSCK.

**Note 5:** This is not a measured value but guaranteed by design.

## 15.0 Switching Characteristics AC Specs DP83901A Note: All Timing is Preliminary (Continued)

Transmit Timing (End of Packet)

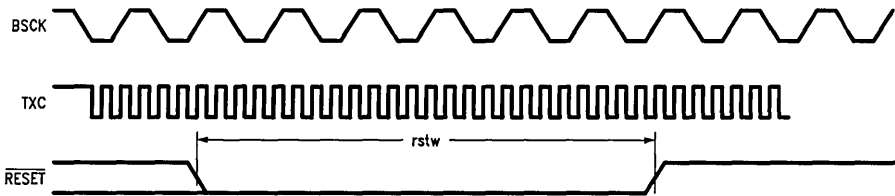


TL/F/10469-47

TRANSMIT SPECIFICATIONS (End of Packet)

Symbol	Parameter	Min	Max	Units
$t_{TOH}$	Transmit Output High before Idle (Half Step)	200		ns
$t_{TOI}$	Transmit Output Idle Time to $\pm 40$ mV(Half Step)		8000	ns

Reset Timing



TL/F/10469-45

Symbol	Parameter	Min	Max	Units
rstw	Reset Pulse Width (Note 1)	8		BSKC Cycles or TXC Cycles (Note 2)

**Note 1:** The RESET pulse requires the BSKC and TXC be stable. On power up, RESET should not be raised until BSKC and TXC have become stable. Several registers are affected by RESET. Consult the register descriptions for details.

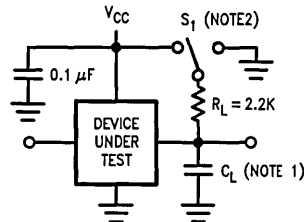
**Note 2:** The slower of BSKC or TXC clocks will determine the minimum time for the RESET signal to be low. TXC is X1 divided by 2.

If BSKC < TXC then RESET = 8 x BSKC  
 If TXC < BSKC then RESET = 8 x TXC

## 16.0 AC Timing Test Conditions

All specifications are valid only if the mandatory isolation is employed and all differential signals are taken to be at the AUI side of the pulse transformer.

- Input Pulse Levels (TTL/CMOS) GND to 3.0V
- Input Rise and Fall Times (TTL/CMOS) 5 ns
- Input and Output Reference Levels (TTL/CMOS) 1.3V
- Input Pulse Levels (Diff.) -350 mV to -1315 mV
- Input and Output Reference Levels (Diff.) 50% Point of the Differential
- TRI-STATE Reference Levels Float ( $\Delta V$ )  $\pm 0.5V$
- Output Load (See Figure Below)



TL/F/10469-48

**Note 1:** 50 pF, includes scope and jig capacitance

**Note 2:** S1 = Open for timing tests for push pull outputs.

S1 = VCC for VOL test.

S1 = GND for VOH test.

S1 = VCC for High Impedance to active low and active low to High Impedance measurements.

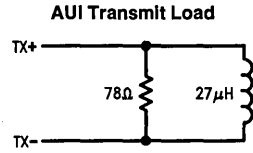
S1 = GND for High Impedance to active high and active high to High Impedance measurements.

**Pin Capacitance**  $T_A = 25^\circ\text{C}$ ,  $f = 1\text{ MHz}$ 

Symbol	Parameter	Typ	Units
$C_{IN}$	Input Capacitance	7	pF
$C_{OUT}$	Output Capacitance	7	pF

**DERATING FACTOR**

Output timings are measured with a purely capacitive load for 50 pF. The following correction factor can be used for other loads:  $C_L \geq 50\text{ pF} + 0.3\text{ ns/pF}$ .



TL/F/10469-49

**Note:** In the above diagram, the TX+ and TX- signals are taken from the AUI side of the isolation (pulse transformer). The pulse transformer used for all testing is the Pulse Engineering PE64103.

# DP8390D/NS32490D NIC Network Interface Controller

## General Description

The DP8390D/NS32490D Network Interface Controller (NIC) is a microCMOS VLSI device designed to ease interfacing with CSMA/CD type local area networks including Ethernet, Thin Ethernet (Cheapernet) and StarLAN. The NIC implements all Media Access Control (MAC) layer functions for transmission and reception of packets in accordance with the IEEE 802.3 Standard. Unique dual DMA channels and an internal FIFO provide a simple yet efficient packet management design. To minimize system parts count and cost, all bus arbitration and memory support logic are integrated into the NIC.

The NIC is the heart of a three chip set that implements the complete IEEE 802.3 protocol and node electronics as shown below. The others include the DP8391 Serial Network Interface (SNI) and the DP8392 Coaxial Transceiver Interface (CTI).

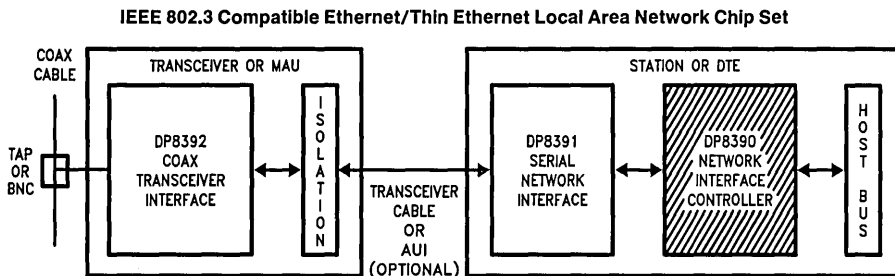
## Features

- Compatible with IEEE 802.3/Ethernet II/Thin Ethernet/StarLAN
- Interfaces with 8-, 16- and 32-bit microprocessor systems
- Implements simple, versatile buffer management
- Requires single 5V supply
- Utilizes low power microCMOS process
- Includes
  - Two 16-bit DMA channels
  - 16-byte internal FIFO with programmable threshold
  - Network statistics storage
- Supports physical, multicast, and broadcast address filtering
- Provides 3 levels of loopback
- Utilizes independent system and network clocks

## Table of Contents

- 1.0 SYSTEM DIAGRAM
- 2.0 BLOCK DIAGRAM
- 3.0 FUNCTIONAL DESCRIPTION
- 4.0 TRANSMIT/RECEIVE PACKET ENCAPSULATION/DECAPSULATION
- 5.0 PIN DESCRIPTIONS
- 6.0 DIRECT MEMORY ACCESS CONTROL (DMA)
- 7.0 PACKET RECEPTION
- 8.0 PACKET TRANSMISSION
- 9.0 REMOTE DMA
- 10.0 INTERNAL REGISTERS
- 11.0 INITIALIZATION PROCEDURES
- 12.0 LOOPBACK DIAGNOSTICS
- 13.0 BUS ARBITRATION AND TIMING
- 14.0 PRELIMINARY ELECTRICAL CHARACTERISTICS
- 15.0 SWITCHING CHARACTERISTICS
- 16.0 PHYSICAL DIMENSIONS

## 1.0 System Diagram



TL/F/8582-1

## 2.0 Block Diagram

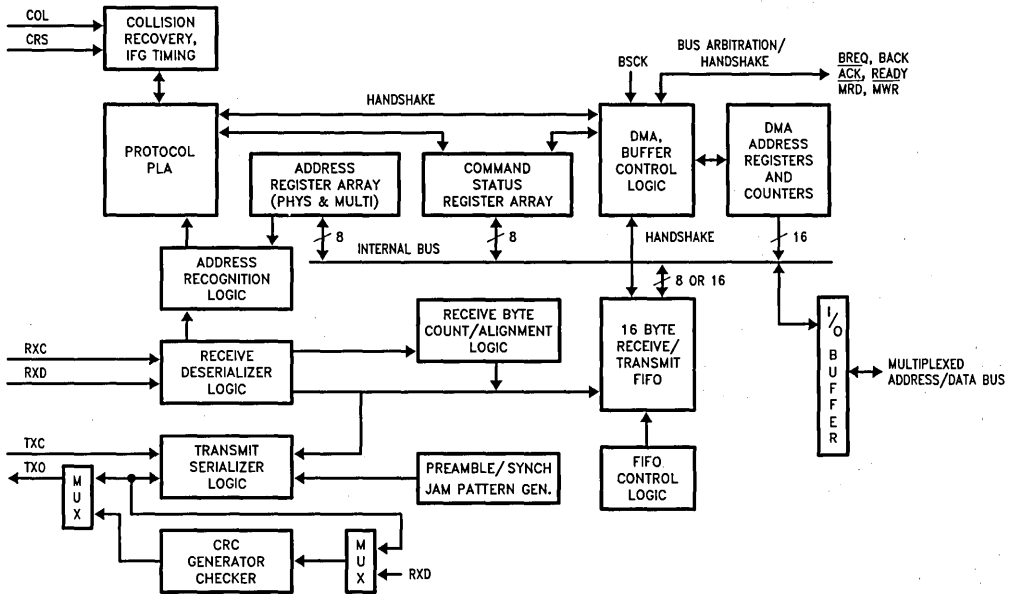


FIGURE 1

TL/F/8592-2

## 3.0 Functional Description

(Refer to Figure 1)

### RECEIVE DESERIALIZER

The Receive Deserializer is activated when the input signal Carrier Sense is asserted to allow incoming bits to be shifted into the shift register by the receive clock. The serial receive data is also routed to the CRC generator/checker. The Receive Deserializer includes a synch detector which detects the SFD (Start of Frame Delimiter) to establish where byte boundaries within the serial bit stream are located. After every eight receive clocks, the byte wide data is transferred to the 16-byte FIFO and the Receive Byte Count is incremented. The first six bytes after the SFD are checked for valid comparison by the Address Recognition Logic. If the Address Recognition Logic does not recognize the packet, the FIFO is cleared.

### CRC GENERATOR/CHECKER

During transmission, the CRC logic generates a local CRC field for the transmitted bit sequence. The CRC encodes all fields after the synch byte. The CRC is shifted out MSB first following the last transmit byte. During reception the CRC logic generates a CRC field from the incoming packet. This local CRC is serially compared to the incoming CRC appended to the end of the packet by the transmitting node. If the local and received CRC match, a specific pattern will be generated and decoded to indicate no data errors. Transmission errors result in a different pattern and are detected, resulting in rejection of a packet.

### TRANSMIT SERIALIZER

The Transmit Serializer reads parallel data from the FIFO and serializes it for transmission. The serializer is clocked by

the transmit clock generated by the Serial Network Interface (DP8391). The serial data is also shifted into the CRC generator/checker. At the beginning of each transmission, the Preamble and Synch Generator append 62 bits of 1,0 preamble and a 1,1 synch pattern. After the last data byte of the packet has been serialized the 32-bit FCS field is shifted directly out of the CRC generator. In the event of a collision the Preamble and Synch generator is used to generate a 32-bit JAM pattern of all 1's

### ADDRESS RECOGNITION LOGIC

The address recognition logic compares the Destination Address Field (first 6 bytes of the received packet) to the Physical address registers stored in the Address Register Array. If any one of the six bytes does not match the pre-programmed physical address, the Protocol Control Logic rejects the packet. All multicast destination addresses are filtered using a hashing technique. (See register description.) If the multicast address indexes a bit that has been set in the filter bit array of the Multicast Address Register Array the packet is accepted, otherwise it is rejected by the Protocol Control Logic. Each destination address is also checked for all 1's which is the reserved broadcast address.

### FIFO AND FIFO CONTROL LOGIC

The NIC features a 16-byte FIFO. During transmission the DMA writes data into the FIFO and the Transmit Serializer reads data from the FIFO and transmits it. During reception the Receive Deserializer writes data into the FIFO and the DMA reads data from the FIFO. The FIFO control logic is used to count the number of bytes in the FIFO so that after a preset level, the DMA can begin a bus access and write/read data to/from the FIFO before a FIFO underflow/overflow occurs.

### 3.0 Functional Description (Continued)

Because the NIC must buffer the Address field of each incoming packet to determine whether the packet matches its Physical Address Registers or maps to one of its Multicast Registers, the first local DMA transfer does not occur until 8 bytes have accumulated in the FIFO.

To assure that there is no overwriting of data in the FIFO, the FIFO logic flags a FIFO overrun as the 13th byte is written into the FIFO; this effectively shortens the FIFO to 13 bytes. In addition, the FIFO logic operates differently in Byte Mode than in Word Mode. In Byte Mode, a threshold is indicated when the  $n + 1$  byte has entered the FIFO; thus, with an 8-byte threshold, the NIC issues Bus Request (BREQ) when the 9th byte has entered the FIFO. For Word Mode, BREQ is not generated until the  $n + 2$  bytes have entered the FIFO. Thus, with a 4 word threshold (equivalent to an 8-byte threshold), BREQ is issued when the 10th byte has entered the FIFO.

#### PROTOCOL PLA

The protocol PLA is responsible for implementing the IEEE 802.3 protocol, including collision recovery with random backoff. The Protocol PLA also formats packets during transmission and strips preamble and synch during reception.

#### DMA AND BUFFER CONTROL LOGIC

The DMA and Buffer Control Logic is used to control two 16-bit DMA channels. During reception, the Local DMA stores packets in a receive buffer ring, located in buffer memory. During transmission the Local DMA uses programmed pointer and length registers to transfer a packet from local buffer memory to the FIFO. A second DMA channel is used as a slave DMA to transfer data between the local buffer memory and the host system. The Local DMA and Remote DMA are internally arbitrated, with the Local DMA channel having highest priority. Both DMA channels use a common external bus clock to generate all required bus timing. External arbitration is performed with a standard bus request, bus acknowledge handshake protocol.

### 4.0 Transmit/Receive Packet Encapsulation/Decapsulation

A standard IEEE 802.3 packet consists of the following fields: preamble, Start of Frame Delimiter (SFD), destination address, source address, length, data, and Frame Check Sequence (FCS). The typical format is shown in *Figure 2*. The packets are Manchester encoded and decoded by the DP8391 SNI and transferred serially to the NIC using NRZ data with a clock. All fields are of fixed length except for the data field. The NIC generates and appends the preamble, SFD and FCS field during transmission. The Preamble and SFD fields are stripped during reception. (The CRC is passed through to buffer memory during reception.)

#### PREAMBLE AND START OF FRAME DELIMITER (SFD)

The Manchester encoded alternating 1,0 preamble field is used by the SNI (DP8391) to acquire bit synchronization with an incoming packet. When transmitted each packet contains 62 bits of alternating 1,0 preamble. Some of this preamble will be lost as the packet travels through the network. The preamble field is stripped by the NIC. Byte alignment is performed with the Start of Frame Delimiter (SFD) pattern which consists of two consecutive 1's. The NIC does not treat the SFD pattern as a byte, it detects only the

two bit pattern. This allows any preceding preamble within the SFD to be used for phase locking.

#### DESTINATION ADDRESS

The destination address indicates the destination of the packet on the network and is used to filter unwanted packets from reaching a node. There are three types of address formats supported by the NIC: physical, multicast, and broadcast. The physical address is a unique address that corresponds only to a single node. All physical addresses have an MSB of "0". These addresses are compared to the internally stored physical address registers. Each bit in the destination address must match in order for the NIC to accept the packet. Multicast addresses begin with an MSB of "1". The DP8390D filters multicast addresses using a standard hashing algorithm that maps all multicast addresses into a 6-bit value. This 6-bit value indexes a 64-bit array that filters the value. If the address consists of all 1's it is a broadcast address, indicating that the packet is intended for all nodes. A promiscuous mode allows reception of all packets: the destination address is not required to match any filters. Physical, broadcast, multicast, and promiscuous address modes can be selected.

#### SOURCE ADDRESS

The source address is the physical address of the node that sent the packet. Source addresses cannot be multicast or broadcast addresses. This field is simply passed to buffer memory.

#### LENGTH FIELD

The 2-byte length field indicates the number of bytes that are contained in the data field of the packet. This field is not interpreted by the NIC.

#### DATA FIELD

The data field consists of anywhere from 46 to 1500 bytes. Messages longer than 1500 bytes need to be broken into multiple packets. Messages shorter than 46 bytes will require appending a pad to bring the data field to the minimum length of 46 bytes. If the data field is padded, the number of valid data bytes is indicated in the length field. **The NIC does not strip or append pad bytes for short packets, or check for oversize packets.**

#### FCS FIELD

The Frame Check Sequence (FCS) is a 32-bit CRC field calculated and appended to a packet during transmission to allow detection of errors when a packet is received. During reception, error free packets result in a specific pattern in the CRC generator. Packets with improper CRC will be rejected. The AUTODIN II ( $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$ ) polynomial is used for the CRC calculations.

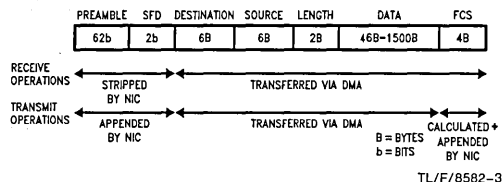
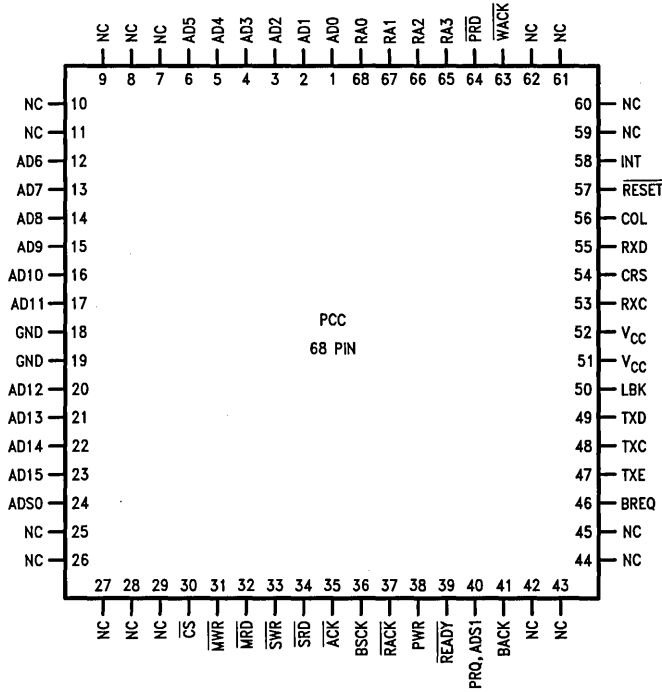


FIGURE 2



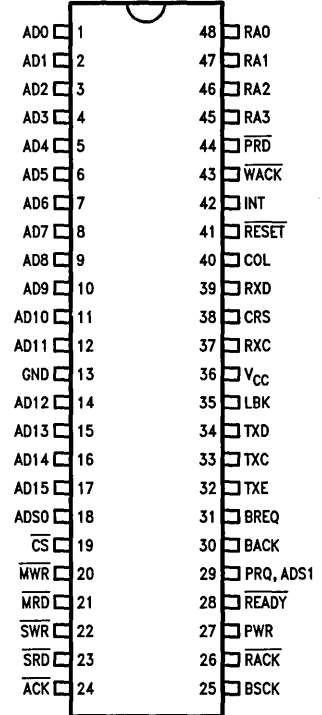
# Connection Diagrams

Plastic Chip Carrier



TL/F/8582-5

Dual-In-Line Package



TL/F/8582-4

Order Number DP8390DN or DP8390DV  
See NS Package Number N48A or V68A

## 5.0 Pin Descriptions

### BUS INTERFACE PINS

Symbol	DIP Pin No	Function	Description
AD0-AD15	1-12 14-17	I/O,Z	<b>MULTIPLEXED ADDRESS/DATA BUS:</b> <ul style="list-style-type: none"> <li>Register Access, with DMA inactive, <math>\overline{CS}</math> low and <math>\overline{ACK}</math> returned from NIC, pins AD0-AD7 are used to read/write register data. AD8-AD15 float during I/O transfers. <math>\overline{SRD}</math>, <math>\overline{SWR}</math> pins are used to select direction of transfer.</li> <li>Bus Master with BACK input asserted. During t1 of memory cycle AD0-AD15 contain address. During t2, t3, t4 AD0-AD15 contain data (word transfer mode). During t2, t3, t4 AD0-AD7 contain data, AD8-AD15 contain address (byte transfer mode). Direction of transfer is indicated by NIC on <math>\overline{MWR}</math>, <math>\overline{MRD}</math> lines.</li> </ul>
ADS0	18	I/O,Z	<b>ADDRESS STROBE 0</b> <ul style="list-style-type: none"> <li>Input with DMA inactive and <math>\overline{CS}</math> low, latches RA0-RA3 inputs on falling edge. If high, data present on RA0-RA3 will flow through latch.</li> <li>Output when Bus Master, latches address bits (A0-A15) to external memory during DMA transfers.</li> </ul>



## 5.0 Pin Descriptions (Continued)

### BUS INTERFACE PINS (Continued)

Symbol	DIP Pin No	Function	Description
$\overline{CS}$	19	I	<b>CHIP SELECT:</b> Chip Select places controller in slave mode for $\mu P$ access to internal registers. Must be valid through data portion of bus cycle. RA0–RA3 are used to select the internal register. SWR and SRD select direction of data transfer.
$\overline{MWR}$	20	O,Z	<b>MASTER WRITE STROBE:</b> Strobe for DMA transfers, active low during write cycles (t2, t3, tw) to buffer memory. Rising edge coincides with the presence of valid output data. TRI-STATE® until BACK asserted.
$\overline{MRD}$	21	O,Z	<b>MASTER READ STROBE:</b> Strobe for DMA transfers, active during read cycles (t2, t3, tw) to buffer memory. Input data must be valid on rising edge of $\overline{MRD}$ . TRI-STATE until BACK asserted.
$\overline{SWR}$	22	I	<b>SLAVE WRITE STROBE:</b> Strobe from CPU to write an internal register selected by RA0–RA3.
$\overline{SRD}$	23	I	<b>SLAVE READ STROBE:</b> Strobe from CPU to read an internal register selected by RA0–RA3.
$\overline{ACK}$	24	O	<b>ACKNOWLEDGE:</b> Active low when NIC grants access to CPU. Used to insert WAIT states to CPU until NIC is synchronized for a register read or write operation.
RA0–RA3	45–48	I	<b>REGISTER ADDRESS:</b> These four pins are used to select a register to be read or written. The state of these inputs is ignored when the NIC is not in slave mode ( $\overline{CS}$ high).
$\overline{PRD}$	44	O	<b>PORT READ:</b> Enables data from external latch onto local bus during a memory write cycle to local memory (remote write operation). This allows asynchronous transfer of data from the system memory to local memory.
$\overline{WACK}$	43	I	<b>WRITE ACKNOWLEDGE:</b> Issued from system to NIC to indicate that data has been written to the external latch. The NIC will begin a write cycle to place the data in local memory.
INT	42	O	<b>INTERRUPT:</b> Indicates that the NIC requires CPU attention after reception transmission or completion of DMA transfers. The interrupt is cleared by writing to the ISR. All interrupts are maskable.
$\overline{RESET}$	41	I	<b>RESET:</b> Reset is active low and places the NIC in a reset mode immediately, no packets are transmitted or received by the NIC until STA bit is set. Affects Command Register, Interrupt Mask Register, Data Configuration Register and Transmit Configuration Register. The NIC will execute reset within 10 BUSK cycles.
BREQ	31	O	<b>BUS REQUEST:</b> Bus Request is an active high signal used to request the bus for DMA transfers. This signal is automatically generated when the FIFO needs servicing.
BACK	30	I	<b>BUS ACKNOWLEDGE:</b> Bus Acknowledge is an active high signal indicating that the CPU has granted the bus to the NIC. If immediate bus access is desired, BREQ should be tied to BACK. <b>Tying BACK to <math>V_{CC}</math> will result in a deadlock.</b>
PRQ, ADS1	29	O,Z	<b>PORT REQUEST/ADDRESS STROBE 1</b> <ul style="list-style-type: none"> <li>• <b>32-BIT MODE:</b> If LAS is set in the Data Configuration Register, this line is programmed as ADS1. It is used to strobe addresses A16–A31 into external latches. (A16–A31 are the fixed addresses stored in RSAR0, RSAR1.) ADS1 will remain at TRI-STATE until BACK is received.</li> <li>• <b>16-BIT MODE:</b> If LAS is not set in the Data Configuration Register, this line is programmed as PRQ and is used for Remote DMA Transfers. In this mode PRQ will be a standard logic output.</li> </ul> <b>NOTE: This line will power up as TRI-STATE until the Data Configuration Register is programmed.</b>
READY	28	I	<b>READY:</b> This pin is set high to insert wait states during a DMA transfer. The NIC will sample this signal at t3 during DMA transfers.

## 5.0 Pin Descriptions (Continued)

### BUS INTERFACE PINS (Continued)

Symbol	DIP Pin No	Function	Description
$\overline{PWR}$	27	O	<b>PORT WRITE:</b> Strobe used to latch data from the NIC into external latch for transfer to host memory during Remote Read transfers. The rising edge of $\overline{PWR}$ coincides with the presence of valid data on the local bus.
$\overline{RACK}$	26	I	<b>READ ACKNOWLEDGE:</b> Indicates that the system DMA or host CPU has read the data placed in the external latch by the NIC. The NIC will begin a read cycle to update the latch.
BSCK	25	I	This clock is used to establish the period of the DMA memory cycle. Four clock cycles (t1, t2, t3, t4) are used per DMA cycle. DMA transfers can be extended by one BSCK increments using the $\overline{READY}$ input.

### NETWORK INTERFACE PINS

COL	40	I	<b>COLLISION DETECT:</b> This line becomes active when a collision has been detected on the coaxial cable. During transmission this line is monitored after preamble and synch have been transmitted. At the end of each transmission this line is monitored for CD heartbeat.
RXD	39	I	<b>RECEIVE DATA:</b> Serial NRZ data received from the ENDEC, clocked into the NIC on the rising edge of RXC.
CRS	38	I	<b>CARRIER SENSE:</b> This signal is provided by the ENDEC and indicates that carrier is present. This signal is active high.
RXC	37	I	<b>RECEIVE CLOCK:</b> Re-synchronized clock from the ENDEC used to clock data from the ENDEC into the NIC.
LBK	35	O	<b>LOOPBACK:</b> This output is set high when the NIC is programmed to perform a loopback through the StarLAN ENDEC.
TXD	34	O	<b>TRANSMIT DATA:</b> Serial NRZ Data output to the ENDEC. The data is valid on the rising edge of TXC.
TXC	33	I	<b>TRANSMIT CLOCK:</b> This clock is used to provide timing for internal operation and to shift bits out of the transmit serializer. TXC is nominally a 1 MHz clock provided by the ENDEC.
TXE	32	O	<b>TRANSMIT ENABLE:</b> This output becomes active when the first bit of the packet is valid on TXD and goes low after the last bit of the packet is clocked out of TXD. This signal connects directly to the ENDEC. This signal is active high.

### POWER

$V_{CC}$	36		+ 5V DC is required. It is suggested that a decoupling capacitor be connected between these pins. It is essential to provide a path to ground for the GND pin with the lowest possible impedance.
GND	13		

## 6.0 Direct Memory Access Control (DMA)

The DMA capabilities of the NIC greatly simplify use of the DP8390D in typical configurations. The local DMA channel transfers data between the FIFO and memory. On transmission, the packet is DMA'd from memory to the FIFO in bursts. Should a collision occur (up to 15 times), the packet is retransmitted with no processor intervention. On reception, packets are DMAed from the FIFO to the receive buffer ring (as explained below).

A remote DMA channel is also provided on the NIC to accomplish transfers between a buffer memory and system memory. The two DMA channels can alternatively be combined to form a single 32-bit address with 8- or 16-bit data.

### DUAL DMA CONFIGURATION

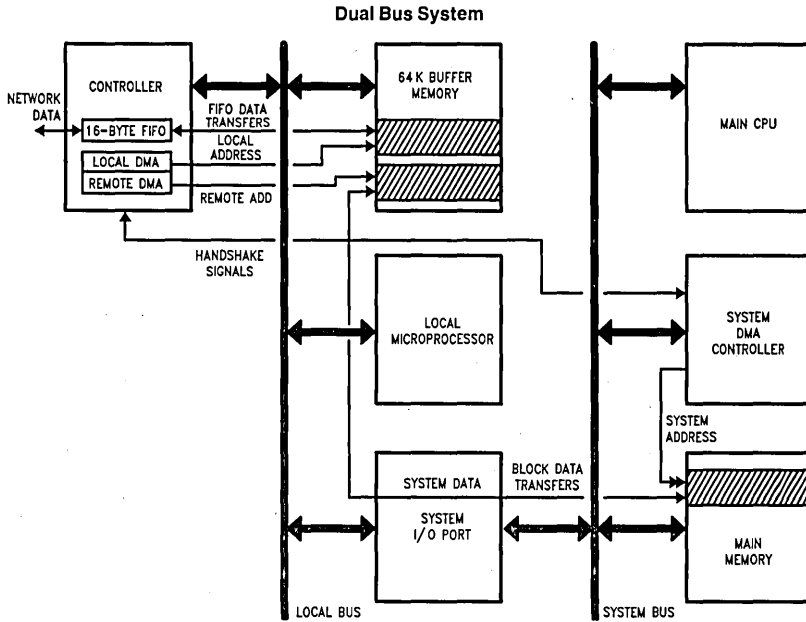
An example configuration using both the local and remote DMA channels is shown below. Network activity is isolated

on a local bus, where the NIC's local DMA channel performs burst transfers between the buffer memory and the NIC's FIFO. The Remote DMA transfers data between the buffer memory and the host memory via a bidirectional I/O port. The Remote DMA provides local addressing capability and is used as a slave DMA by the host. Host side addressing must be provided by a host DMA or the CPU. The NIC allows Local and Remote DMA operations to be interleaved.

### SINGLE CHANNEL DMA OPERATION

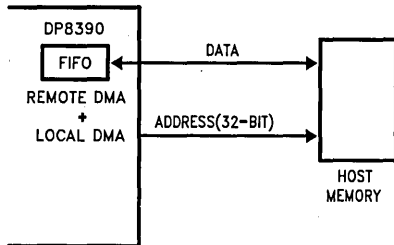
If desirable, the two DMA channels can be combined to provide a 32-bit DMA address. The upper 16 bits of the 32-bit address are static and are used to point to a 64k byte (or 32k word) page of memory where packets are to be received and transmitted.

## 6.0 Direct Memory Access Control (DMA) (Continued)



TL/F/8582-55

### 32-Bit DMA Operation

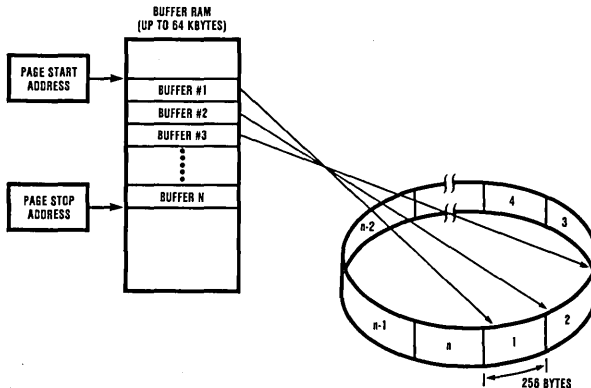


TL/F/8582-6

## 7.0 Packet Reception

The Local DMA receive channel uses a Buffer Ring Structure comprised of a series of contiguous fixed length 256 byte (128 word) buffers for storage of received packets. The location of the Receive Buffer Ring is programmed in two registers, a Page Start and a Page Stop Register. Ethernet packets consist of a distribution of shorter link control packets and longer data packets, the 256 byte buffer length provides a good compromise between short packets and longer packets to most efficiently use memory. In addition these buffers provide memory resources for storage of back-to-back packets in loaded networks. The assignment of buffers

### NIC Receive Buffer Ring



TL/F/8582-7

## 7.0 Packet Reception (Continued)

for storing packets is controlled by Buffer Management Logic in the NIC. The Buffer Management Logic provides three basic functions: linking receive buffers for long packets, recovery of buffers when a packet is rejected, and recirculation of buffer pages that have been read by the host.

At initialization, a portion of the 64k byte (or 32k word) address space is reserved for the receive buffer ring. Two eight bit registers, the Page Start Address Register (PSTART) and the Page Stop Address Register (PSTOP) define the physical boundaries of where the buffers reside. The NIC treats the list of buffers as a logical ring; whenever the DMA address reaches the Page Stop Address, the DMA is reset to the Page Start Address.

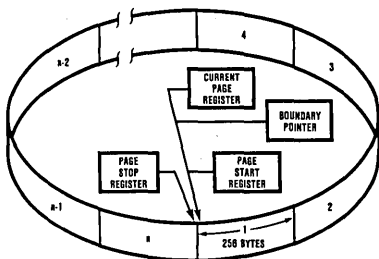
### INITIALIZATION OF THE BUFFER RING

Two static registers and two working registers control the operation of the Buffer Ring. These are the Page Start Register, Page Stop Register (both described previously), the Current Page Register and the Boundary Pointer Register. The Current Page Register points to the first buffer used to store a packet and is used to restore the DMA for writing status to the Buffer Ring or for restoring the DMA address in the event of a Runt packet, a CRC, or Frame Alignment error. The Boundary Register points to the first packet in the Ring not yet read by the host. If the local DMA address ever reaches the Boundary, reception is aborted. The Boundary Pointer is also used to initialize the Remote DMA for removing a packet and is advanced when a packet is removed. A simple analogy to remember the function of these registers is that the Current Page Register acts as a Write Pointer and the Boundary Pointer acts as a Read Pointer.

**Note 1:** At initialization, the Page Start Register value should be loaded into both the Current Page Register and the Boundary Pointer Register.

**Note 2:** The Page Start Register must not be initialized to 00H.

Receive Buffer Ring At Initialization

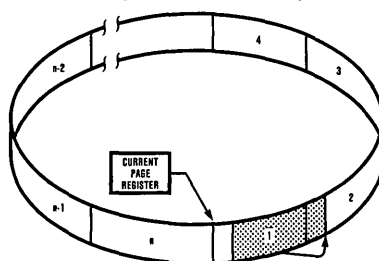


TL/F/8582-30

### BEGINNING OF RECEPTION

When the first packet begins arriving the NIC begins storing the packet at the location pointed to by the Current Page

Linking Receive Buffer Pages

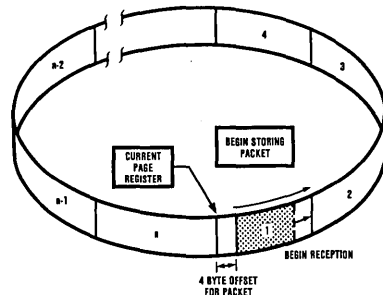


- 1) Check for = to PSTOP
- 2) Check for = to Boundary

TL/F/8582-32

Register. An offset of 4 bytes is saved in this first buffer to allow room for storing receive status corresponding to this packet.

Received Packet Enters Buffer Pages



TL/F/8582-31

### LINKING RECEIVE BUFFER PAGES

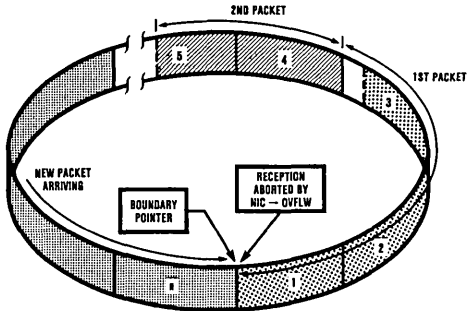
If the length of the packet exhausts the first 256 byte buffer, the DMA performs a forward link to the next buffer to store the remainder of the packet. For a maximal length packet the buffer logic will link six buffers to store the entire packet. Buffers cannot be skipped when linking, a packet will always be stored in contiguous buffers. Before the next buffer can be linked, the Buffer Management Logic performs two comparisons. The first comparison tests for equality between the DMA address of the next buffer and the contents of the Page Stop Register. If the buffer address equals the Page Stop Register, the buffer management logic will restore the DMA to the first buffer in the Receive Buffer Ring value programmed in the Page Start Address Register. The second comparison tests for equality between the DMA address of the next buffer address and the contents of the Boundary Pointer Register. If the two values are equal the reception is aborted. The Boundary Pointer Register can be used to protect against overwriting any area in the receive buffer ring that has not yet been read. When linking buffers, buffer management will never cross this pointer, effectively avoiding any overwrites. If the buffer address does not match either the Boundary Pointer or Page Stop Address, the link to the next buffer is performed.

### Linking Buffers

Before the DMA can enter the next contiguous 256 byte buffer, the address is checked for equality to PSTOP and to the Boundary Pointer. If neither are reached, the DMA is allowed to use the next buffer.

## 7.0 Packet Reception (Continued)

### Received Packet Aborted if It Hits Boundary Pointer



TL/F/8582-8

### Buffer Ring Overflow

If the Buffer Ring has been filled and the DMA reaches the Boundary Pointer Address, reception of the incoming packet will be aborted by the NIC. Thus, the packets previously received and still contained in the Ring will not be destroyed.

In a heavily loaded network environment the local DMA may be disabled, preventing the NIC from buffering packets from the network. To guarantee this will not happen, a software reset must be issued during all Receive Buffer Ring overflows (indicated by the OVW bit in the Interrupt Status Register). **The following procedure is required to recover from a Receiver Buffer Ring Overflow.**

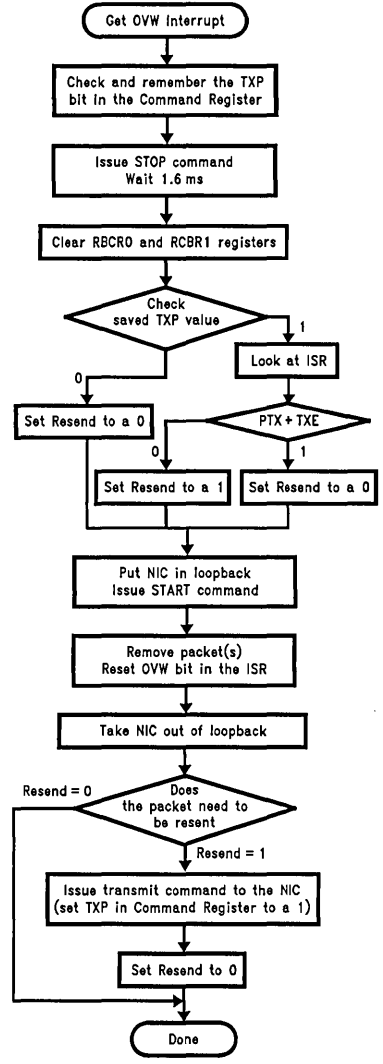
If this routine is not adhered to, the NIC may act in an unpredictable manner. It should also be noted that it is not permissible to service an overflow interrupt by continuing to empty packets from the receive buffer without implementing the prescribed overflow routine. A flow chart of the NIC's overflow routine can be found at the right.

**Note:** It is necessary to define a variable in the driver, which will be called "Resend".

1. Read and store the value of the TXP bit in the NIC's Command Register.
2. Issue the STOP command to the NIC. This is accomplished by setting the STP bit in the NIC's Command Register. Writing 21H to the Command Register will stop the NIC.

**Note:** If the STP is set when a transmission is in progress, the RST bit may not be set. In this case, the NIC is guaranteed to be reset after the longest packet time (1500 bytes = 1.2 ms). For the DP8390D (but not for the DP8390B), the NIC will be reset within 2 microseconds after the STP bit is set and Loopback mode 1 is programmed.

3. Wait for at least 1.6 ms. Since the NIC will complete any transmission or reception that is in progress, it is necessary to time out for the maximum possible duration of an Ethernet transmission or reception. By waiting 1.6 ms this is achieved with some guard band added. Previously, it was recommended that the RST bit of the Interrupt Status Register be polled to insure that the pending transmission or reception is completed. This bit is not a reliable indicator and subsequently should be ignored.
4. Clear the NIC's Remote Byte Count registers (RBCR0 and RBCR1).



TL/F/8582-95

Overflow Routine Flow Chart

5. Read the stored value of the TXP bit from step 1, above. If this value is a 0, set the "Resend" variable to a 0 and jump to step 6. If this value is a 1, read the NIC's Interrupt Status Register. If either the Packet Transmitted bit (PTX) or Transmit Error bit (TXE) is set to a 1, set the "Resend" variable to a 0 and jump to step 6. If neither of these bits is set, place a 1 in the "Resend" variable and jump to step 6. This step determines if there was a transmission in progress when the stop command was issued in step 2. If there was a transmission in progress, the NIC's ISR is read to determine whether or not the packet was recognized by the NIC. If neither the PTX nor TXE bit was set,

## 7.0 Packet Reception (Continued)

then the packet will essentially be lost and re-transmitted only after a time-out takes place in the upper level software. By determining that the packet was lost at the driver level, a transmit command can be reissued to the NIC once the overflow routine is completed (as in step 11). Also, it is possible for the NIC to defer indefinitely, when it is stopped on a busy network. Step 5 also alleviates this problem. Step 5 is essential and should not be omitted from the overflow routine, in order for the NIC to operate correctly.

6. Place the NIC in either mode 1 or mode 2 loopback. This can be accomplished by setting bits D2 and D1, of the Transmit Configuration Register, to "0,1" or "1,0", respectively.
7. Issue the START command to the NIC. This can be accomplished by writing 22H to the Command Register. This is necessary to activate the NIC's Remote DMA channel.
8. Remove one or more packets from the receive buffer ring.
9. Reset the overwrite warning (OVW, overflow) bit in the Interrupt Status Register.
10. Take the NIC out of loopback. This is done by writing the Transmit Configuration Register with the value it contains during normal operation. (Bits D2 and D1 should both be programmed to 0.)
11. If the "Resend" variable is set to a 1, reset the "Resend" variable and reissue the transmit command. This is done by writing a value of 26H to the Command Register. If the "Resend" variable is 0, nothing needs to be done.

**Note:** If Remote DMA is not being used, the NIC does not need to be started before packets can be removed from the receive buffer ring. Hence, step 8 could be done before step 7.

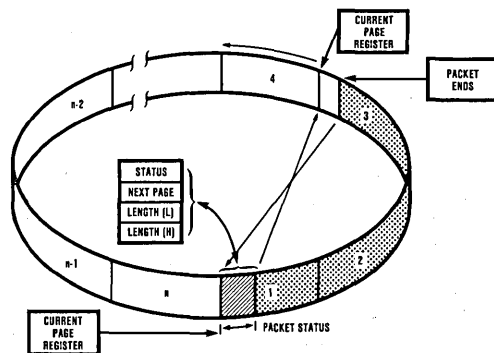
### END OF PACKET OPERATIONS

At the end of the packet the NIC determines whether the received packet is to be accepted or rejected. It either branches to a routine to store the Buffer Header or to another routine that recovers the buffers used to store the packet.

### SUCCESSFUL RECEPTION

If the packet is successfully received as shown, the DMA is restored to the first buffer used to store the packet (pointed

#### Termination of Received Packet—Packet Accepted



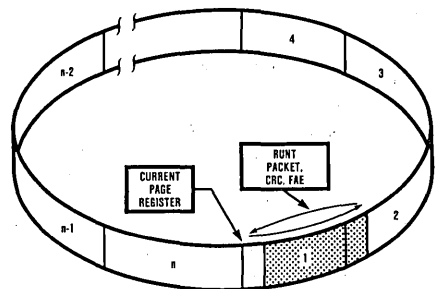
TL/F/8582-10

to by the Current Page Register). The DMA then stores the Receive Status, a Pointer to where the next packet will be stored (Buffer 4) and the number of received bytes. Note that the remaining bytes in the last buffer are discarded and reception of the next packet begins on the next empty 256-byte buffer boundary. The Current Page Register is then initialized to the next available buffer in the Buffer Ring. (The location of the next buffer had been previously calculated and temporarily stored in an internal scratchpad register.)

### BUFFER RECOVERY FOR REJECTED PACKETS

If the packet is a runt packet or contains CRC or Frame Alignment errors, it is rejected. The buffer management logic resets the DMA back to the first buffer page used to store the packet (pointed to by CURR), recovering all buffers that had been used to store the rejected packet. This operation will not be performed if the NIC is programmed to accept either runt packets or packets with CRC or Frame Alignment errors. The received CRC is always stored in buffer memory after the last byte of received data for the packet.

#### Termination of Received Packet—Packet Rejected



TL/F/8582-13

### Error Recovery

If the packet is rejected as shown, the DMA is restored by the NIC by reprogramming the DMA starting address pointed to by the Current Page Register.

### REMOVING PACKETS FROM THE RING

Packets are removed from the ring using the Remote DMA or an external device. When using the Remote DMA the Send Packet command can be used. This programs the Remote DMA to automatically remove the received packet pointed to by the Boundary Pointer. At the end of the transfer, the NIC moves the Boundary Pointer, freeing additional buffers for reception. The Boundary Pointer can also be moved manually by programming the Boundary Register. Care should be taken to keep the Boundary Pointer at least one buffer behind the Current Page Pointer.

The following is a suggested method for maintaining the Receive Buffer Ring pointers.

1. At initialization, set up a software variable (next\_pkt) to indicate where the next packet will be read. At the beginning of each Remote Read DMA operation, the value of next\_pkt will be loaded into RSAR0 and RSAR1.
2. When initializing the NIC set:
  - BNDRY = PSTART
  - CURR = PSTART + 1
  - next\_pkt = PSTART + 1

## 7.0 Packet Reception (Continued)

3. After a packet is DMAed from the Receive Buffer Ring, the Next Page Pointer (second byte in NIC buffer header) is used to update BNDRY and next\_pkt.

next\_pkt = Next Page Pointer

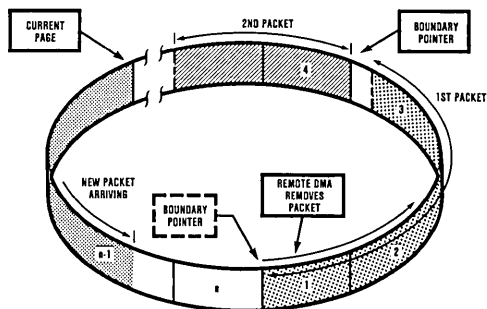
BNDRY = Next Page Pointer - 1

If BNDRY < PSTART then BNDRY = PSTOP - 1

Note the size of the Receive Buffer Ring is reduced by one 256-byte buffer; this will not, however, impede the operation of the NIC.

In StarLAN applications using bus clock frequencies greater than 4 MHz, the NIC does not update the buffer header information properly because of the disparity between the network and bus clock speeds. The lower byte count is copied twice into the third and fourth locations of the buffer header and the upper byte count is not written. The upper byte count, however, can be calculated from the current next page pointer (second byte in the buffer header) and the previous next page pointer (stored in memory by the CPU). The following routine calculates the upper byte count and allows StarLAN applications to be insensitive to bus clock speeds. Next\_pkt is defined similarly as above.

### 1st Received Packet Removed By Remote DMA



TL/F/8582-57

upper byte count = next page pointer - next\_pkt - 1

if (upper byte count) < 0 then

upper byte count = (PSTOP - next\_pkt) +

(next page pointer - PSTART) - 1

if (lower byte count) > 0 fch then

upper byte count = upper byte count + 1

### STORAGE FORMAT FOR RECEIVED PACKETS

The following diagrams describe the format for how received packets are placed into memory by the local DMA channel. These modes are selected in the Data Configuration Register.

#### Storage Format

AD15	AD8	AD7	AD0
Next Packet Pointer	Receive Status		
Receive Byte Count 1	Receive Byte Count 0		
Byte 2	Byte 1		

BOS = 0, WTS = 1 in Data Configuration Register.

This format used with Series 32000 808X type processors.

AD15	AD8	AD7	AD0
Next Packet Pointer		Receive Status	
Receive Byte Count 0		Receive Byte Count 1	
Byte 1		Byte 2	

BOS = 1, WTS = 1 in Data Configuration Register.

This format used with 68000 type processors.

Note: The Receive Byte Count ordering remains the same for BOS=0 or 1.

AD7	AD0
Receive Status	
Next Packet Pointer	
Receive Byte Count 0	
Receive Byte Count 1	
Byte 0	
Byte 1	

BOS = 0, WTS = 0 in Data Configuration Register.

This format used with general 8-bit CPUs.

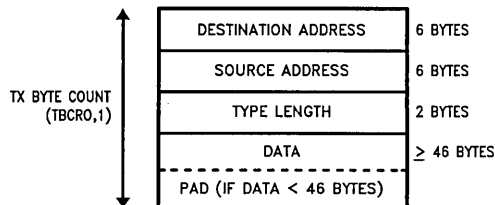
## 8.0 Packet Transmission

The Local DMA is also used during transmission of a packet. Three registers control the DMA transfer during transmission, a Transmit Page Start Address Register (TPSR) and the Transmit Byte Count Registers (TBCR0,1). When the NIC receives a command to transmit the packet pointed to by these registers, buffer memory data will be moved into the FIFO as required during transmission. The NIC will generate and append the preamble, synch and CRC fields.

### TRANSMIT PACKET ASSEMBLY

The NIC requires a contiguous assembled packet with the format shown. The transmit byte count includes the Destination Address, Source Address, Length Field and Data. It does not include preamble and CRC. When transmitting data smaller than 46 bytes, the packet must be padded to a minimum size of 64 bytes. The programmer is responsible for adding and stripping pad bytes.

#### General Transmit Packet Format



TL/F/8582-58

## 8.0 Packet Transmission (Continued)

### TRANSMISSION

Prior to transmission, the TPSR (Transmit Page Start Register) and TBCR0, TBCR1 (Transmit Byte Count Registers) must be initialized. To initiate transmission of the packet the TXP bit in the Command Register is set. The Transmit Status Register (TSR) is cleared and the NIC begins to pre-fetch transmit data from memory (unless the NIC is currently receiving). If the interframe gap has timed out the NIC will begin transmission.

### CONDITIONS REQUIRED TO BEGIN TRANSMISSION

In order to transmit a packet, the following three conditions must be met:

1. The Interframe Gap Timer has timed out the first 6.4  $\mu$ s of the Interframe Gap (See appendix for Interframe Gap Flowchart)
2. At least one byte has entered the FIFO. (This indicates that the burst transfer has been started)
3. If the NIC had collided, the backoff timer has expired.

In typical systems the NIC has already prefetched the first burst of bytes before the 6.4  $\mu$ s timer expires. The time during which NIC transmits preamble can also be used to load the FIFO.

**Note:** If carrier sense is asserted before a byte has been loaded into the FIFO, the NIC will become a receiver.

### COLLISION RECOVERY

During transmission, the Buffer Management logic monitors the transmit circuitry to determine if a collision has occurred. If a collision is detected, the Buffer Management logic will reset the FIFO and restore the Transmit DMA pointers for retransmission of the packet. The COL bit will be set in the TSR and the NCR (Number of Collisions Register) will be incremented. If 15 retransmissions each result in a collision the transmission will be aborted and the ABT bit in the TSR will be set.

**Note:** NCR reads as zeroes if excessive collisions are encountered.

### TRANSMIT PACKET ASSEMBLY FORMAT

The following diagrams describe the format for how packets must be assembled prior to transmission for different byte ordering schemes. The various formats are selected in the Data Configuration Register.

D15	D8 D7	D0
DA1		DA0
DA3		DA2
DA5		DA4
SA1		DA0
SA3		DA2
SA5		DA4
T/L1		T/L0
DATA 1		DATA 0

BOS = 0, WTS = 1 in Data Configuration Register.

This format is used with Series 32000, 808X type processors.

D15	D8 D7	D0
DA0		DA1
DA2		DA3
DA4		DA5
SA0		SA1
SA2		SA3
SA4		SA5
T/L0		T/L1
DATA 0		DATA 1

BOS = 1, WTS = 1 in Data Configuration Register.

This format is used with 68000 type processors.

D7	D0
	DA0
	DA1
	DA2
	DA3
	DA4
	DA5
	SA0
	SA1
	SA2
	SA3

BOS = 0, WTS = 0 in Data Configuration Register.

This format is used with general 8-bit CPUs.

**Note:** All examples above will result in a transmission of a packet in order of DA0, DA1, DA2, DA3 . . . bits within each byte will be transmitted least significant bit first.

DA = Destination Address

SA = Source Address

T/L = Type/Length Field

## 9.0 Remote DMA

The Remote DMA channel is used to both assemble packets for transmission, and to remove received packets from the Receive Buffer Ring. It may also be used as a general purpose slave DMA channel for moving blocks of data or commands between host memory and local buffer memory. There are three modes of operation, Remote Write, Remote Read, or Send Packet.

Two register pairs are used to control the Remote DMA, a Remote Start Address (RSAR0, RSAR1) and a Remote Byte Count (RBCR0, RBCR1) register pair. The Start Address Register pair points to the beginning of the block to be moved while the Byte Count Register pair is used to indicate the number of bytes to be transferred. Full handshake logic is provided to move data between local buffer memory and a bidirectional I/O port.



## 9.0 Remote DMA (Continued)

### REMOTE WRITE

A Remote Write transfer is used to move a block of data from the host into local buffer memory. The Remote DMA will read data from the I/O port and sequentially write it to local buffer memory beginning at the Remote Start Address. The DMA Address will be incremented and the Byte Counter will be decremented after each transfer. The DMA is terminated when the Remote Byte Count Register reaches a count of zero.

### REMOTE READ

A Remote Read transfer is used to move a block of data from local buffer memory to the host. The Remote DMA will sequentially read data from the local buffer memory, beginning at the Remote Start Address, and write data to the I/O port. The DMA Address will be incremented and the Byte Counter will be decremented after each transfer. The DMA is terminated when the Remote Byte Count Register reaches zero.

### REMOTE DMA WRITE

#### Setting PRQ Using the Remote Read

Under certain conditions the NIC's bus state machine may issue /MWR and /PRD before PRQ for the first DMA transfer of a Remote Write Command. If this occurs this could cause data corruption, or cause the remote DMA count to be different from the main CPU count causing the system to "lock up".

To prevent this condition when implementing a Remote DMA Write, the Remote DMA Write command should first be preceded by a Remote DMA Read command to insure that the PRQ signal is asserted before the NIC starts its port read cycle. The reason for this is that the state machine that asserts PRQ runs independently of the state machine that controls the DMA signals. The DMA machine assumes that PRQ is asserted, but actually may not be. To remedy this situation, a single Remote Read cycle should be inserted before the actual DMA Write Command is given. This will ensure that PRQ is asserted when the Remote DMA Write is subsequently executed. This single Remote Read cycle is

called a "dummy Remote Read." In order for the dummy Remote Read cycle to operate correctly, the Start Address should be programmed to a known, safe location in the buffer memory space, and the Remote Byte Count should be programmed to a value greater than 1. This will ensure that the master read cycle is performed safely, eliminating the possibility of data corruption.

#### Remote Write with High Speed Buses

When implementing the Remote DMA Write solution in previous section with high speed buses and CPU's, timing problems may cause the system to hang. Therefore additional considerations are required.

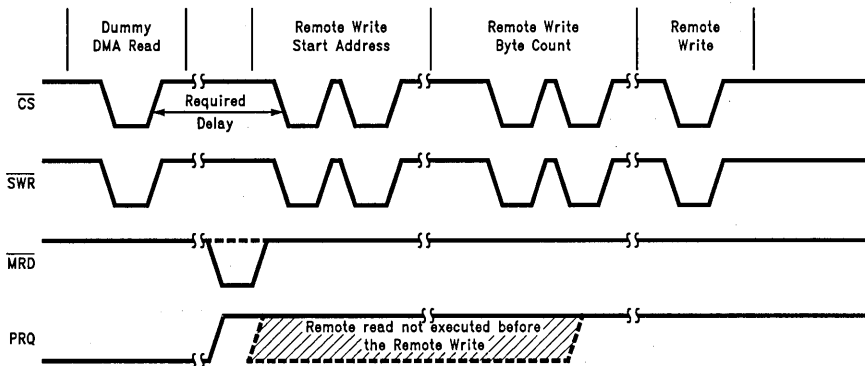
The problem occurs when the system can execute the dummy Remote Read and then start the Remote Write before the NIC has had a chance to execute the Remote Read. If this happens the PRQ signal will not get set, and the Remote Byte Count and Remote Start Address for the Remote Write operation could be corrupted. This is shown by the hatched waveforms in the timing diagram below. The execution of the Remote Read can be delayed by the local DMA operations (particularly during end-of-packet processing).

To ensure the dummy Remote Read does execute, a delay must be inserted between writing the Remote Read Command, and starting to write the Remote Write Start Address. (This time is designated in figure below by the delay arrows.) The recommended method to avoid this problem is, after the Remote Read command is given, to poll both bytes of the Current Remote DMA Address Registers. When the address has incremented, PRQ has been set. Software should recognize this and then start the Remote Write.

An additional caution for high speed systems is that the polling must follow guidelines specified at the end of Section 13. That is, there must be at least 4 bus clocks between chip selects. (For example, when BSCK = 20 MHz, then this time should be 200 ns.)

The general flow for executing a Remote Write is:

1. Set Remote Byte Count to a value > 1 and Remote Start Address to unused RAM (one location before the transmit start address is usually a safe location).



Timing Diagram for Dummy Remote Read

TL/F/8582-98

Note: The dashed lines indicate incorrect timing.

## 9.0 Remote DMA (Continued)

2. Issue the "dummy" Remote Read command.
3. Read the Current Remote DMA Address (CRDA) (both bytes).
4. Compare to previous CRDA value if different go to 6.
5. Delay and jump to 3.
6. Set up for the Remote Write command, by setting the Remote Byte Count and the Remote Start Address (note that if the Remote Byte Count in step 1 can be set to the transmit byte count plus one, and the Remote Start Address to one less, these will now be incremented to the correct values.)
7. Issue the Remote Write command.

### FIFO AND BUS OPERATIONS

#### Overview

To accommodate the different rates at which data comes from (or goes to) the network and goes to (or comes from) the system memory, the NIC contains a 16-byte FIFO for buffering data between the bus and the media. The FIFO threshold is programmable, allowing filling (or emptying) the FIFO at different rates. When the FIFO has filled to its programmed threshold, the local DMA channel transfers these bytes (or words) into local memory. It is crucial that the local DMA is given access to the bus within a minimum bus latency time; otherwise a FIFO underrun (or overrun) occurs.

To understand FIFO underruns or overruns, there are two causes which produce this condition—

- 1) the bus latency is so long that the FIFO has filled (or emptied) from the network before the local DMA has serviced the FIFO.
- 2) the bus latency or bus data rate has slowed the throughput of the local DMA to point where it is slower than the network data rate (10 Mb/s). This second condition is also dependent upon DMA clock and word width (byte wide or word wide).

The worst case condition ultimately limits the overall bus latency which the NIC can tolerate.

#### FIFO Underrun and Transmit Enable

During transmission, if a FIFO underrun occurs, the Transmit enable (TXE) output may remain high (active). Generally, this will cause a very large packet to be transmitted onto the network. The jabber feature of the transceiver will terminate the transmission, and reset TXE.

To prevent this problem, a properly designed system will not allow FIFO underruns by giving the NIC a bus acknowledge within time shown in the maximum bus latency curves shown and described later.

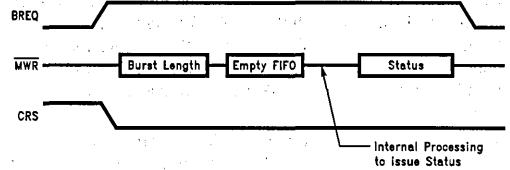
#### FIFO at the Beginning of Receive

At the beginning of reception, the NIC stores entire Address field of each incoming packet in the FIFO to determine whether the packet matches its Physical Address Registers or maps to one of its Multicast Registers. This causes the FIFO to accumulate 8 bytes. Furthermore, there are some synchronization delays in the DMA PLA. Thus, the actual time that BREQ is asserted from the time the Start of Frame Delimiter (SFD) is detected is 7.8  $\mu$ s. This operation affects the bus latencies at 2 and 4 byte thresholds during the first receive BREQ since the FIFO must be filled to 8 bytes (4 words) before issuing a BREQ.

#### FIFO Operation at the End of Receive

When Carrier Sense goes low, the NIC enters its end of packet processing sequence, emptying its FIFO and writing the status information at the beginning of the packet, figure below. This NIC holds onto the bus for the entire sequence. The longest time BREQ may be extended occurs when a packet ends just as the NIC performs its last FIFO burst. The NIC, in this case, performs a programmed burst transfer followed by flushing the remaining bytes in the FIFO, and completes by writing the header information to memory. The following steps occur during this sequence.

- 1) NIC issues BREQ because the FIFO threshold has been reached.
- 2) During the burst, packet ends, resulting in BREQ extended.
- 3) NIC flushes remaining bytes from FIFO.
- 4) NIC performs internal processing to prepare for writing the header.
- 5) NIC writes 4-byte (2-word) header.
- 6) NIC deasserts BREQ.



#### End of Packet Processing

End of Packet Processing (EOPP) times for 10 MHz and 20 MHz have been tabulated in the table below.

**End of Packet Processing Times for Various FIFO Thresholds, Bus Clocks and Transfer Modes**

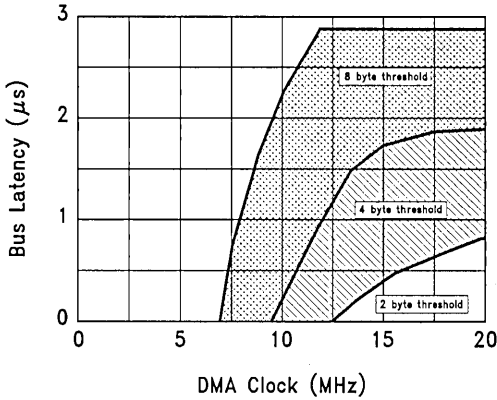
Mode	Threshold	Bus Clock	EOPP
Byte	2 bytes	10 MHz	7.0 $\mu$ s
	4 bytes		8.6 $\mu$ s
	8 bytes		11.0 $\mu$ s
Byte	2 bytes	20 MHz	3.6 $\mu$ s
	4 bytes		4.2 $\mu$ s
	8 bytes		5.0 $\mu$ s
Word	2 bytes	10 MHz	5.4 $\mu$ s
	4 bytes		6.2 $\mu$ s
	8 bytes		7.4 $\mu$ s
Word	2 bytes	20 MHz	3.0 $\mu$ s
	4 bytes		3.2 $\mu$ s
	8 bytes		3.6 $\mu$ s

#### Threshold Detection (Bus Latency)

To assure that no overwriting of data in the FIFO, the FIFO logic flags a FIFO overrun as the 13th byte is written into the FIFO, effectively shortening the FIFO to 13 bytes. The FIFO logic also operates differently in Byte Mode and in Word Mode. In Byte Mode, a threshold is indicated when the n + 1

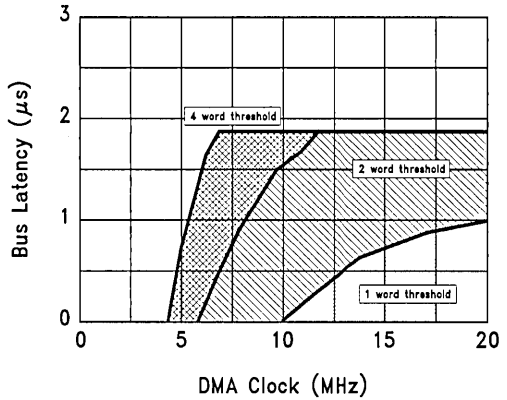
## 9.0 Remote DMA (Continued)

Maximum Bus Latency for Byte Mode



TL/F/8582-98

Maximum Bus Latency for Word Mode



TL/F/8582-99

byte has entered the FIFO; thus, with an 8 byte threshold, the NIC issues Bus Request (BREQ) when the 9th byte has entered the FIFO. For Word Mode, BREQ is not generated until the  $n + 2$  bytes have entered the FIFO. Thus, with a 4 word threshold (equivalent to 8 byte threshold), BREQ is issued when the 10th byte has entered the FIFO. The two graphs, the figures above, indicate the maximum allowable bus latency for Word and Byte transfer modes.

### The FIFO at the Beginning of Transmit

Before transmitting, the NIC performs a prefetch from memory to load the FIFO. The number of bytes prefetched is the programmed FIFO threshold. The next BREQ is not issued until after the NIC actually begins transmitting data, i.e., after SFD. The Transmit Prefetch diagram illustrates this process.

### SEND PACKET COMMAND

The Remote DMA channel can be automatically initialized to transfer a single packet from the Receive Buffer Ring.

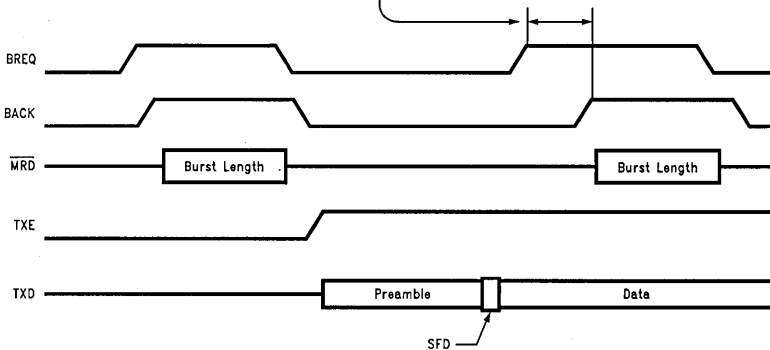
The CPU begins this transfer by issuing a "Send Packet" Command. The DMA will be initialized to the value of the Boundary Pointer Register and the Remote Byte Count Register pair (RBCR0, RBCR1) will be initialized to the value of the Receive Byte Count fields found in the Buffer Header of each packet. After the data is transferred, the Boundary Pointer is advanced to allow the buffers to be used for new receive packets. The Remote Read will terminate when the Byte Count equals zero. The Remote DMA is then prepared to read the next packet from the Receive Buffer Ring. If the DMA pointer crosses the Page Stop Register, it is reset to the Page Start Address. This allows the Remote DMA to remove packets that have wrapped around to the top of the Receive Buffer Ring.

**Note 1:** In order for the NIC to correctly execute the Send Packet Command, the upper Remote Byte Count Register (RBCR1) must first be loaded with 0FH.

**Note 2:** The Send Packet command cannot be used with 68000 type processors.

### Transmit Prefetch Timing

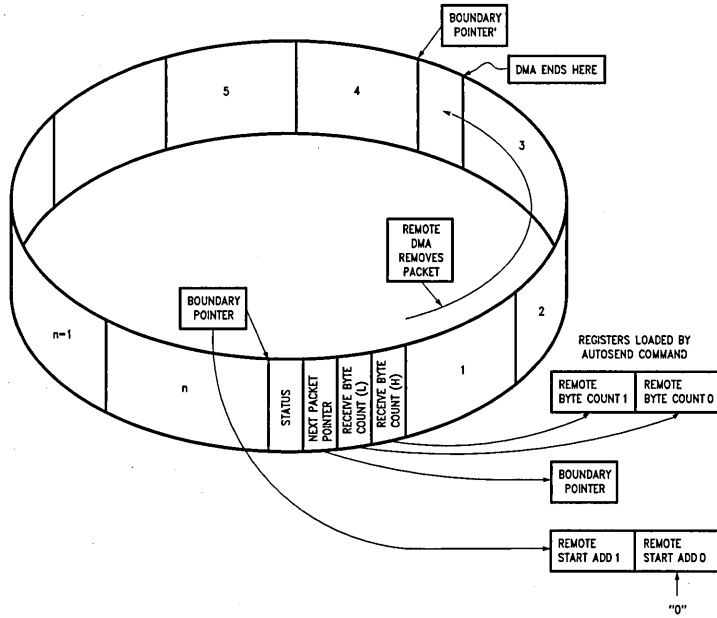
$$\text{Tolerated Bus Latency} = \left[ \left( \text{No. of Bytes Stored in FIFO} \times 800 \right) - 400 \text{ ns} \right] \text{ or } \left( 12 \text{ Bytes} - \text{FIFO Threshold} \right) \text{ whichever is less}$$



TL/F/8582-A0

## 9.0 Remote DMA (Continued)

### Remote DMA Autoinitialization from Buffer Ring

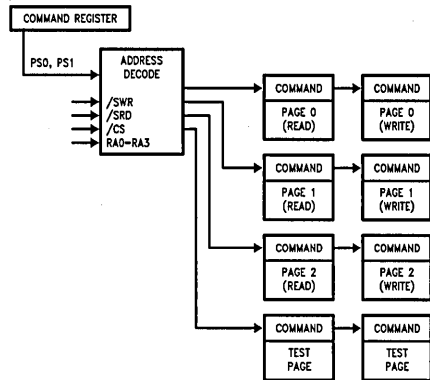


TL/F/8582-59

## 10.0 Internal Registers

All registers are 8-bit wide and mapped into two pages which are selected in the Command Register (PS0, PS1). Pins RA0-RA3 are used to address registers within each page. Page 0 registers are those registers which are commonly accessed during NIC operation while page 1 registers are used primarily for initialization. The registers are partitioned to avoid having to perform two write/read cycles to access commonly used registers.

## 10.1 REGISTER ADDRESS MAPPING



TL/F/8582-60

## 10.0 Internal Registers (Continued)

### 10.2 REGISTER ADDRESS ASSIGNMENTS

#### Page 0 Address Assignments (PS1 = 0, PS0 = 0)

RA0-RA3	RD	WR
00H	Command (CR)	Command (CR)
01H	Current Local DMA Address 0 (CLDA0)	Page Start Register (PSTART)
02H	Current Local DMA Address 1 (CLDA1)	Page Stop Register (PSTOP)
03H	Boundary Pointer (BNRY)	Boundary Pointer (BNRY)
04H	Transmit Status Register (TSR)	Transmit Page Start Address (TPSR)
05H	Number of Collisions Register (NCR)	Transmit Byte Count Register 0 (TBCR0)
06H	FIFO (FIFO)	Transmit Byte Count Register 1 (TBCR1)
07H	Interrupt Status Register (ISR)	Interrupt Status Register (ISR)
08H	Current Remote DMA Address 0 (CRDA0)	Remote Start Address Register 0 (RSAR0)
09H	Current Remote DMA Address 1 (CRDA1)	Remote Start Address Register 1 (RSAR1)
0AH	Reserved	Remote Byte Count Register 0 (RBCR0)
0BH	Reserved	Remote Byte Count Register 1 (RBCR1)
0CH	Receive Status Register (RSR)	Receive Configuration Register (RCR)
0DH	Tally Counter 0 (Frame Alignment Errors) (CNTR0)	Transmit Configuration Register (TCR)
0EH	Tally Counter 1 (CRC Errors) (CNTR1)	Data Configuration Register (DCR)
0FH	Tally Counter 2 (Missed Packet Errors) (CNTR2)	Interrupt Mask Register (IMR)

#### Page 1 Address Assignments (PS1 = 0, PS0 = 1)

RA0-RA3	RD	WR
00H	Command (CR)	Command (CR)
01H	Physical Address Register 0 (PAR0)	Physical Address Register 0 (PAR0)
02H	Physical Address Register 1 (PAR1)	Physical Address Register 1 (PAR1)
03H	Physical Address Register 2 (PAR2)	Physical Address Register 2 (PAR2)
04H	Physical Address Register 3 (PAR3)	Physical Address Register 3 (PAR3)
05H	Physical Address Register 4 (PAR4)	Physical Address Register 4 (PAR4)
06H	Physical Address Register 5 (PAR5)	Physical Address Register 5 (PAR5)
07H	Current Page Register (CURR)	Current Page Register (CURR)
08H	Multicast Address Register 0 (MAR0)	Multicast Address Register 0 (MAR0)
09H	Multicast Address Register 1 (MAR1)	Multicast Address Register 1 (MAR1)
0AH	Multicast Address Register 2 (MAR2)	Multicast Address Register 2 (MAR2)
0BH	Multicast Address Register 3 (MAR3)	Multicast Address Register 3 (MAR3)
0CH	Multicast Address Register 4 (MAR4)	Multicast Address Register 4 (MAR4)
0DH	Multicast Address Register 5 (MAR5)	Multicast Address Register 5 (MAR5)
0EH	Multicast Address Register 6 (MAR6)	Multicast Address Register 6 (MAR6)
0FH	Multicast Address Register 7 (MAR7)	Multicast Address Register 7 (MAR7)

## 10.0 Internal Registers (Continued)

### Page 2 Address Assignments (PS1 = 1, PS0 = 0)

RA0-RA3	RD	WR
00H	Command (CR)	Command (CR)
01H	Page Start Register (PSTART)	Current Local DMA Address 0 (CLDA0)
02H	Page Stop Register (PSTOP)	Current Local DMA Address 1 (CLDA1)
03H	Remote Next Packet Pointer	Remote Next Packet Pointer
04H	Transmit Page Start Address (TPSR)	Reserved
05H	Local Next Packet Pointer	Local Next Packet Pointer
06H	Address Counter (Upper)	Address Counter (Upper)
07H	Address Counter (Lower)	Address Counter (Lower)

RA0-RA3	RD	WR
08H	Reserved	Reserved
09H	Reserved	Reserved
0AH	Reserved	Reserved
0BH	Reserved	Reserved
0CH	Receive Configuration Register (RCR)	Reserved
0DH	Transmit Configuration Register (TCR)	Reserved
0EH	Data Configuration Register (DCR)	Reserved
0FH	Interrupt Mask Register (IMR)	Reserved

**Note:** Page 2 registers should only be accessed for diagnostic purposes. They should not be modified during normal operation.

Page 3 should never be modified.

## 10.0 Internal Registers (Continued)

### 10.3 Register Descriptions

#### COMMAND REGISTER (CR) 00H (READ/WRITE)

The Command Register is used to initiate transmissions, enable or disable Remote DMA operations and to select register pages. To issue a command the microprocessor sets the corresponding bit(s) (RD2, RD1, RD0, TXP). Further commands may be overlapped, but with the following rules: (1) If a transmit command overlaps with a remote DMA operation, bits RD0, RD1, and RD2 must be maintained for the remote DMA command when setting the TXP bit. Note, if a remote DMA command is re-issued when giving the transmit command, the DMA will complete immediately if the remote byte count register have not been re-initialized. (2) If a remote DMA operation overlaps a transmission, RD0, RD1, and RD2 may be written with the desired values and a "0" written to the TXP bit. Writing a "0" to this bit has no effect. (3) A remote write DMA may not overlap remote read operation or visa versa. Either of these operations must either complete or be aborted before the other operation may start. Bits PS1, PS0, RD2, and STP may be set any time.

7	6	5	4	3	2	1	0
PS1	PS0	RD2	RD1	RD0	TXP	STA	STP

Bit	Symbol	Description																								
D0	STP	<p><b>STOP:</b> Software reset command, takes the controller offline, no packets will be received or transmitted. Any reception or transmission in progress will continue to completion before entering the reset state. To exit this state, the STP bit must be reset and the STA bit must be set high. To perform a software reset, this bit should be set high. The software reset has executed only when indicated by the RST bit in the ISR being set to a 1. <b>STP powers up high.</b></p> <p><b>Note:</b> If the NIC has previously been in start mode and the STP is set, both the STP and STA bits will remain set.</p>																								
D1	STA	<p><b>START:</b> This bit is used to activate the NIC after either power up, or when the NIC has been placed in a reset mode by software command or error. <b>STA powers up low.</b></p>																								
D2	TXP	<p><b>TRANSMIT PACKET:</b> This bit must be set to initiate transmission of a packet. TXP is internally reset either after the transmission is completed or aborted. This bit should be set only after the Transmit Byte Count and Transmit Page Start registers have been programmed.</p> <p><b>Note:</b> Before the transmit command is given, the STA bit must be set and the STP bit reset.</p>																								
D3, D4, D5	RD0, RD1, RD2	<p><b>REMOTE DMA COMMAND:</b> These three encoded bits control operation of the Remote DMA channel. RD2 can be set to abort any Remote DMA command in progress. The Remote Byte Count Registers should be cleared when a Remote DMA has been aborted. The Remote Start Addresses are not restored to the starting address if the Remote DMA is aborted.</p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="text-align: left;">RD2</td> <td style="text-align: left;">RD1</td> <td style="text-align: left;">RD0</td> <td></td> </tr> <tr> <td>0</td> <td>0</td> <td>0</td> <td>Not Allowed</td> </tr> <tr> <td>0</td> <td>0</td> <td>1</td> <td>Remote Read</td> </tr> <tr> <td>0</td> <td>1</td> <td>0</td> <td>Remote Write (Note 2)</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> <td>Send Packet</td> </tr> <tr> <td>1</td> <td>X</td> <td>X</td> <td>Abort/Complete Remote DMA (Note 1)</td> </tr> </table> <p><b>Note 1:</b> If a remote DMA operation is aborted and the remote byte count has not decremented to zero, PRQ (pin 29, DIP) will remain high. A read acknowledge (RACK) on a write acknowledge (WACK) will reset PRQ low.</p> <p><b>Note 2:</b> For proper operation of the Remote Write DMA, there are two steps which must be performed before using the Remote Write DMA. The steps are as follows:</p> <ol style="list-style-type: none"> <li>i) Write a non-zero value into RBCR0.</li> <li>ii) Set bits RD2, RD1, RD0 to 0, 0, 1.</li> <li>iii) Set RBCR0, 1 and RSAR0, 1</li> <li>iv) Issue the Remote Write DMA Command (RD2, RD1, RD0 = 0, 1, 0)</li> </ol>	RD2	RD1	RD0		0	0	0	Not Allowed	0	0	1	Remote Read	0	1	0	Remote Write (Note 2)	0	1	1	Send Packet	1	X	X	Abort/Complete Remote DMA (Note 1)
RD2	RD1	RD0																								
0	0	0	Not Allowed																							
0	0	1	Remote Read																							
0	1	0	Remote Write (Note 2)																							
0	1	1	Send Packet																							
1	X	X	Abort/Complete Remote DMA (Note 1)																							
D6, D7	PS0, PS1	<p><b>PAGE SELECT:</b> These two encoded bits select which register page is to be accessed with addresses RA0-3.</p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="text-align: left;">PS1</td> <td style="text-align: left;">PS0</td> <td></td> </tr> <tr> <td>0</td> <td>0</td> <td>Register Page 0</td> </tr> <tr> <td>0</td> <td>1</td> <td>Register Page 1</td> </tr> <tr> <td>1</td> <td>0</td> <td>Register Page 2</td> </tr> <tr> <td>1</td> <td>1</td> <td>Reserved</td> </tr> </table>	PS1	PS0		0	0	Register Page 0	0	1	Register Page 1	1	0	Register Page 2	1	1	Reserved									
PS1	PS0																									
0	0	Register Page 0																								
0	1	Register Page 1																								
1	0	Register Page 2																								
1	1	Reserved																								

## 10.0 Internal Registers (Continued)

### 10.3 Register Descriptions (Continued)

#### INTERRUPT STATUS REGISTER (ISR) 07H (READ/WRITE)

This register is accessed by the host processor to determine the cause of an interrupt. Any interrupt can be masked in the Interrupt Mask Register (IMR). Individual interrupt bits are cleared by writing a "1" into the corresponding bit of the ISR. The INT signal is active as long as any unmasked signal is set, and will not go low until all unmasked bits in this register have been cleared. The ISR must be cleared after power up by writing it with all 1's.

7	6	5	4	3	2	1	0
RST	RDC	CNT	OVW	TXE	RXE	PTX	PRX

Bit	Symbol	Description
D0	PRX	<b>PACKET RECEIVED:</b> Indicates packet received with no errors.
D1	PTX	<b>PACKET TRANSMITTED:</b> Indicates packet transmitted with no errors.
D2	RXE	<b>RECEIVE ERROR:</b> Indicates that a packet was received with one or more of the following errors: —CRC Error —Frame Alignment Error —FIFO Overrun —Missed Packet
D3	TXE	<b>TRANSMIT ERROR:</b> Set when packet transmitted with one or more of the following errors: —Excessive Collisions —FIFO Underrun
D4	OVW	<b>OVERWRITE WARNING:</b> Set when receive buffer ring storage resources have been exhausted. (Local DMA has reached Boundary Pointer).
D5	CNT	<b>COUNTER OVERFLOW:</b> Set when MSB of one or more of the Network Tally Counters has been set.
D6	RDC	<b>REMOTE DMA COMPLETE:</b> Set when Remote DMA operation has been completed.
D7	RST	<b>RESET STATUS:</b> Set when NIC enters reset state and cleared when a Start Command is issued to the CR. This bit is also set when a Receive Buffer Ring overflow occurs and is cleared when one or more packets have been removed from the ring. Writing to this bit has no effect. <b>NOTE:</b> This bit does not generate an interrupt, it is merely a status indicator.

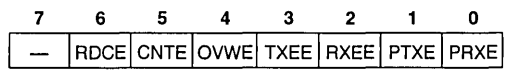


## 10.0 Internal Registers (Continued)

### 10.3 Register Descriptions (Continued)

#### INTERRUPT MASK REGISTER (IMR) 0FH (WRITE)

The Interrupt Mask Register is used to mask interrupts. Each interrupt mask bit corresponds to a bit in the Interrupt Status Register (ISR). If an interrupt mask bit is set an interrupt will be issued whenever the corresponding bit in the ISR is set. If any bit in the IMR is set low, an interrupt will not occur when the bit in the ISR is set. **The IMR powers up all zeroes.**



Bit	Symbol	Description
D0	PRXE	<b>PACKET RECEIVED INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when packet received.
D1	PTXE	<b>PACKET TRANSMITTED INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when packet is transmitted.
D2	RXEE	<b>RECEIVE ERROR INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when packet received with error.
D3	TXEE	<b>TRANSMIT ERROR INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when packet transmission results in error.
D4	OVWE	<b>OVERWRITE WARNING INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when Buffer Management Logic lacks sufficient buffers to store incoming packet.
D5	CNTE	<b>COUNTER OVERFLOW INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when MSB of one or more of the Network Statistics counters has been set.
D6	RDCE	<b>DMA COMPLETE INTERRUPT ENABLE</b> 0: Interrupt Disabled 1: Enables Interrupt when Remote DMA transfer has been completed.
D7	reserved	reserved

## 10.0 Internal Registers (Continued)

### 10.3 Register Descriptions (Continued)

#### DATA CONFIGURATION REGISTER (DCR) 0EH (WRITE)

This Register is used to program the NIC for 8- or 16-bit memory interface, select byte ordering in 16-bit applications and establish FIFO thresholds. **The DCR must be initialized prior to loading the Remote Byte Count Registers. LAS is set on power up.**

7	6	5	4	3	2	1	0
—	FT1	FT0	ARM	LS	LAS	BOS	WTS

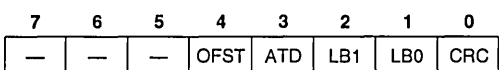
Bit	Symbol	Description																									
D0	WTS	<p><b>WORD TRANSFER SELECT</b>                      0: Selects byte-wide DMA transfers                      1: Selects word-wide DMA transfers</p> <p>; WTS establishes byte or word transfers for both Remote and Local DMA transfers</p> <p><b>Note:</b> When word-wide mode is selected, up to 32k words are addressable; A0 remains low.</p>																									
D1	BOS	<p><b>BYTE ORDER SELECT</b>                      0: MS byte placed on AD15–AD8 and LS byte on AD7–AD0. (32000, 8086)                      1: MS byte placed on AD7–AD0 and LS byte on AD15–AD8. (68000)</p> <p>; Ignored when WTS is low</p>																									
D2	LAS	<p><b>LONG ADDRESS SELECT</b>                      0: Dual 16-bit DMA mode                      1: Single 32-bit DMA mode</p> <p>; When LAS is high, the contents of the Remote DMA registers RSAR0,1 are issued as A16–A31 Power up high.</p>																									
D3	LS	<p><b>LOOPBACK SELECT</b>                      0: Loopback mode selected. Bits D1, D2 of the TCR must also be programmed for Loopback operation.                      1: Normal Operation.</p>																									
D4	AR	<p><b>AUTO-INITIALIZE REMOTE</b>                      0: Send Command not executed, all packets removed from Buffer Ring under program control.                      1: Send Command executed, Remote DMA auto-initialized to remove packets from Buffer Ring.</p> <p><b>Note:</b> Send Command cannot be used with 68000 type processors.</p>																									
D5, D6	FT0, FT1	<p><b>FIFO THRESHHOLD SELECT:</b> Encoded FIFO threshold. Establishes point at which bus is requested when filling or emptying the FIFO. During reception, the FIFO threshold indicates the number of bytes (or words) the FIFO has filled serially from the network before bus request (BREQ) is asserted.</p> <p><b>Note:</b> FIFO threshold setting determines the DMA burst length.</p> <p style="text-align: center;"><b>RECEIVE THRESHOLDS</b></p> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 10%;"></td> <td style="width: 10%;">FT1</td> <td style="width: 10%;">FT0</td> <td style="width: 20%;">Word Wide</td> <td style="width: 20%;">Byte Wide</td> </tr> <tr> <td></td> <td>0</td> <td>0</td> <td>1 Word</td> <td>2 Bytes</td> </tr> <tr> <td></td> <td>0</td> <td>1</td> <td>2 Words</td> <td>4 Bytes</td> </tr> <tr> <td></td> <td>1</td> <td>0</td> <td>4 Words</td> <td>8 Bytes</td> </tr> <tr> <td></td> <td>1</td> <td>1</td> <td>6 Words</td> <td>12 Bytes</td> </tr> </table> <p>During transmission, the FIFO threshold indicates the numer of bytes (or words) the FIFO has filled from the Local DMA before BREQ is asserted. Thus, the transmission threshold is 16 bytes less the receive threshold.</p>		FT1	FT0	Word Wide	Byte Wide		0	0	1 Word	2 Bytes		0	1	2 Words	4 Bytes		1	0	4 Words	8 Bytes		1	1	6 Words	12 Bytes
	FT1	FT0	Word Wide	Byte Wide																							
	0	0	1 Word	2 Bytes																							
	0	1	2 Words	4 Bytes																							
	1	0	4 Words	8 Bytes																							
	1	1	6 Words	12 Bytes																							

## 10.0 Internal Registers (Continued)

### 10.3 Register Descriptions (Continued)

#### TRANSMIT CONFIGURATION REGISTER (TCR) 0DH (WRITE)

The transmit configuration establishes the actions of the transmitter section of the NIC during transmission of a packet on the network. LB1 and LB0 which select loopback mode power up as 0.



Bit	Symbol	Description																				
D0	CRC	<p><b>INHIBIT CRC</b></p> <p>0: CRC appended by transmitter                      1: CRC inhibited by transmitter                      ; In loopback mode CRC can be enabled or disabled to test the CRC logic.</p>																				
D1, D2	LB0, LB1	<p><b>ENCODED LOOPBACK CONTROL:</b> These encoded configuration bits set the type of loopback that is to be performed. Note that loopback in mode 2 sets the LPBK pin high, this places the SNI in loopback mode and that D3 of the DCR must be set to zero for loopback operation.</p> <table style="margin-left: 40px; border-collapse: collapse;"> <tr> <td></td> <td style="text-align: center;">LB1</td> <td style="text-align: center;">LB0</td> <td></td> </tr> <tr> <td>Mode 0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td>Normal Operation (LPBK = 0)</td> </tr> <tr> <td>Mode 1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td>Internal Loopback (LPBK = 0)</td> </tr> <tr> <td>Mode 2</td> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td>External Loopback (LPBK = 1)</td> </tr> <tr> <td>Mode 3</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td>External Loopback (LPBK = 0)</td> </tr> </table>		LB1	LB0		Mode 0	0	0	Normal Operation (LPBK = 0)	Mode 1	0	1	Internal Loopback (LPBK = 0)	Mode 2	1	0	External Loopback (LPBK = 1)	Mode 3	1	1	External Loopback (LPBK = 0)
	LB1	LB0																				
Mode 0	0	0	Normal Operation (LPBK = 0)																			
Mode 1	0	1	Internal Loopback (LPBK = 0)																			
Mode 2	1	0	External Loopback (LPBK = 1)																			
Mode 3	1	1	External Loopback (LPBK = 0)																			
D3	ATD	<p><b>AUTO TRANSMIT DISABLE:</b> This bit allows another station to disable the NIC's transmitter by transmission of a particular multicast packet. The transmitter can be re-enabled by resetting this bit or by reception of a second particular multicast packet.</p> <p>0: Normal Operation                      1: Reception of multicast address hashing to bit 62 disables transmitter, reception of multicast address hashing to bit 63 enables transmitter.</p>																				
D4	OFST	<p><b>COLLISION OFFSET ENABLE:</b> This bit modifies the backoff algorithm to allow prioritization of nodes.</p> <p>0: Backoff Logic implements normal algorithm.                      1: Forces Backoff algorithm modification to 0 to <math>2^{\min(3+n,10)}</math> slot times for first three collisions, then follows standard backoff. (For first three collisions station has higher average backoff delay making a low priority mode.)</p>																				
D5	reserved	reserved																				
D6	reserved	reserved																				
D7	reserved	reserved																				

## 10.0 Internal Registers (Continued)

### 10.3 Register Descriptions (Continued)

#### TRANSMIT STATUS REGISTER (TSR) 04H (READ)

This register records events that occur on the media during transmission of a packet. It is cleared when the next transmission is initiated by the host. All bits remain low unless the event that corresponds to a particular bit occurs during transmission. Each transmission should be followed by a read of this register. The contents of this register are not specified until after the first transmission.

7	6	5	4	3	2	1	0
OWC	CDH	FU	CRS	ABT	COL	—	PTX

Bit	Symbol	Description
D0	PTX	<b>PACKET TRANSMITTED:</b> Indicates transmission without error. (No excessive collisions or FIFO underrun) (ABT = "0", FU = "0").
D1	reserved	reserved
D2	COL	<b>TRANSMIT COLLIDED:</b> Indicates that the transmission collided at least once with another station on the network. The number of collisions is recorded in the Number of Collisions Registers (NCR).
D3	ABT	<b>TRANSMIT ABORTED:</b> Indicates the NIC aborted transmission because of excessive collisions. (Total number of transmissions including original transmission attempt equals 16).
D4	CRS	<b>CARRIER SENSE LOST:</b> This bit is set when carrier is lost during transmission of the packet. Carrier Sense is monitored from the end of Preamble/Synch until TXEN is dropped. Transmission is not aborted on loss of carrier.
D5	FU	<b>FIFO UNDERRUN:</b> If the NIC cannot gain access of the bus before the FIFO empties, this bit is set. Transmission of the packet will be aborted.
D6	CDH	<b>CD HEARTBEAT:</b> Failure of the transceiver to transmit a collision signal after transmission of a packet will set this bit. The Collision Detect (CD) heartbeat signal must commence during the first 6.4 $\mu$ s of the Interframe Gap following a transmission. In certain collisions, the CD Heartbeat bit will be set even though the transceiver is not performing the CD heartbeat test.
D7	OWC	<b>OUT OF WINDOW COLLISION:</b> Indicates that a collision occurred after a slot time (51.2 $\mu$ s). Transmissions rescheduled as in normal collisions.

## 10.0 Internal Registers (Continued)

### 10.3 Register Descriptions (Continued)

#### RECEIVE CONFIGURATION REGISTER (RCR) OCH (WRITE)

This register determines operation of the NIC during reception of a packet and is used to program what types of packets to accept.

7	6	5	4	3	2	1	0
—	—	MON	PRO	AM	AB	AR	SEP

Bit	Symbol	Description
D0	SEP	<p><b>SAVE ERRORED PACKETS</b>                      0: Packets with receive errors are rejected.                      1: Packets with receive errors are accepted. Receive errors are CRC and Frame Alignment errors.</p>
D1	AR	<p><b>ACCEPT RUNT PACKETS:</b> This bit allows the receiver to accept packets that are smaller than 64 bytes. The packet must be at least 8 bytes long to be accepted as a runt.                      0: Packets with fewer than 64 bytes rejected.                      1: Packets with fewer than 64 bytes accepted.</p>
D2	AB	<p><b>ACCEPT BROADCAST:</b> Enables the receiver to accept a packet with an all 1's destination address.                      0: Packets with broadcast destination address rejected.                      1: Packets with broadcast destination address accepted.</p>
D3	AM	<p><b>ACCEPT MULTICAST:</b> Enables the receiver to accept a packet with a multicast address, all multicast addresses must pass the hashing array.                      0: Packets with multicast destination address not checked.                      1: Packets with multicast destination address checked.</p>
D4	PRO	<p><b>PROMISCUOUS PHYSICAL:</b> Enables the receiver to accept all packets with a physical address.                      0: Physical address of node must match the station address programmed in PAR0–PAR5.                      1: All packets with physical addresses accepted.</p>
D5	MON	<p><b>MONITOR MODE:</b> Enables the receiver to check addresses and CRC on incoming packets without buffering to memory. The Missed Packet Tally counter will be incremented for each recognized packet.                      0: Packets buffered to memory.                      1: Packets checked for address match, good CRC and Frame Alignment but not buffered to memory.</p>
D6	reserved	reserved
D7	reserved	reserved

**Note:** D2 and D3 are "OR'd" together, i.e., if D2 and D3 are set the NIC will accept broadcast and multicast addresses as well as its own physical address. To establish full promiscuous mode, bits D2, D3, and D4 should be set. In addition the multicast hashing array must be set to all 1's in order to accept all multicast addresses.

## 10.0 Internal Registers (Continued)

### 10.3 Register Descriptions (Continued)

#### RECEIVE STATUS REGISTER (RSR) OCH (READ)

This register records status of the received packet, including information on errors and the type of address match, either physical or multicast. The contents of this register are written to buffer memory by the DMA after reception of a good packet. If packets with errors are to be saved the receive status is written to memory at the head of the erroneous packet if an erroneous packet is received. If packets with errors are to be rejected the RSR will not be written to memory. The contents will be cleared when the next packet arrives. CRC errors, Frame Alignment errors and missed packets are counted internally by the NIC which relinquishes the Host from reading the RSR in real time to record errors for Network Management Functions. The contents of this register are not specified until after the first reception.

7	6	5	4	3	2	1	0
DFR	DIS	PHY	MPA	FO	FAE	CRC	PRX

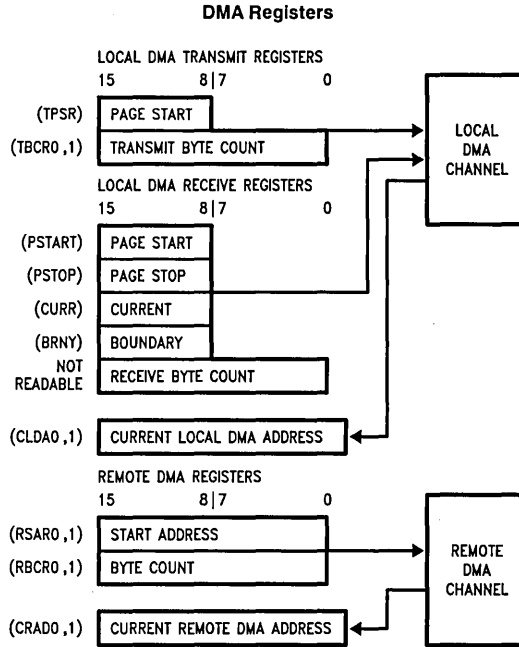
Bit	Symbol	Description
D0	PRX	<b>PACKET RECEIVED INTACT:</b> Indicates packet received without error. (Bits CRC, FAE, FO, and MPA are zero for the received packet.)
D1	CRC	<b>CRC ERROR:</b> Indicates packet received with CRC error. Increments Tally Counter (CNTR1). This bit will also be set for Frame Alignment errors.
D2	FAE	<b>FRAME ALIGNMENT ERROR:</b> Indicates that the incoming packet did not end on a byte boundary and the CRC did not match at last byte boundary. Increments Tally Counter (CNTR0).
D3	FO	<b>FIFO OVERRUN:</b> This bit is set when the FIFO is not serviced causing overflow during reception. Reception of the packet will be aborted.
D4	MPA	<b>MISSED PACKET:</b> Set when packet intended for node cannot be accepted by NIC because of a lack of receive buffers or if the controller is in monitor mode and did not buffer the packet to memory. Increments Tally Counter (CNTR2).
D5	PHY	<b>PHYSICAL/MULTICAST ADDRESS:</b> Indicates whether received packet had a physical or multicast address type. 0: Physical Address Match 1: Multicast/Broadcast Address Match
D6	DIS	<b>RECEIVER DISABLED:</b> Set when receiver disabled by entering Monitor mode. Reset when receiver is re-enabled when exiting Monitor mode.
D7	DFR	<b>DEFERRING:</b> Set when CRS or COL inputs are active. If the transceiver has asserted the CD line as a result of the jabber, this bit will stay set indicating the jabber condition.

**Note:** Following coding applies to CRC and FAE bits

FAE/CRC	Type of Error
0 0	No Error (Good CRC and <6 Dribble Bits)
0 1	CRC Error
1 0	Illegal, will not occur
1 1	Frame Alignment Error and CRC Error

# 10.0 Internal Registers (Continued)

## 10.4 DMA REGISTERS



TL/F/8582-61

The DMA Registers are partitioned into three groups; Transmit, Receive and Remote DMA Registers. The Transmit registers are used to initialize the Local DMA Channel for transmission of packets while the Receive Registers are used to initialize the Local DMA Channel for packet Reception. The Page Stop, Page Start, Current and Boundary Registers are used by the Buffer Management Logic to supervise the Receive Buffer Ring. The Remote DMA Registers are used to initialize the Remote DMA.

**Note:** In the figure above, registers are shown as 8 or 16 bits wide. Although some registers are 16-bit internal registers, all registers are accessed as 8-bit registers. Thus the 16-bit Transmit Byte Count Register is broken into two 8-bit registers, TBCR0 and TBCR1. Also TPSR, PSTART, PSTOP, CURR and BRNY only check or control the upper 8 bits of address information on the bus. Thus they are shifted to positions 15-8 in the diagram above.

### 10.5 TRANSMIT DMA REGISTERS

#### TRANSMIT PAGE START REGISTER (TPSR)

This register points to the assembled packet to be transmitted. Only the eight higher order addresses are specified since all transmit packets are assembled on 256-byte page boundaries. The bit assignment is shown below. The values placed in bits D7-D0 will be used to initialize the higher order address (A8-A15) of the Local DMA for transmission. The lower order bits (A7-A0) are initialized to zero.

Bit Assignment

	7	6	5	4	3	2	1	0
TPSR	A15	A14	A13	A12	A11	A10	A9	A8

(A7-A0 Initialized to zero)

#### TRANSMIT BYTE COUNT REGISTER 0,1 (TBCR0, TBCR1)

These two registers indicate the length of the packet to be transmitted in bytes. The count must include the number of

bytes in the source, destination, length and data fields. The maximum number of transmit bytes allowed is 64k bytes. The NIC will not truncate transmissions longer than 1500 bytes. The bit assignment is shown below:

	7	6	5	4	3	2	1	0
TBCR1	L15	L14	L13	L12	L11	L10	L9	L8

	7	6	5	4	3	2	1	0
TBCR0	L7	L6	L5	L4	L3	L2	L1	L0

### 10.6 LOCAL DMA RECEIVE REGISTERS

#### PAGE START STOP REGISTERS (PSTART, PSTOP)

The Page Start and Page Stop Registers program the starting and stopping address of the Receive Buffer Ring. Since the NIC uses fixed 256-byte buffers aligned on page boundaries only the upper eight bits of the start and stop address are specified.

#### PSTART, PSTOP bit assignment

	7	6	5	4	3	2	1	0
PSTART, PSTOP	A15	A14	A13	A12	A11	A10	A9	A8

#### BOUNDARY (BRNY) REGISTER

This register is used to prevent overflow of the Receive Buffer Ring. Buffer management compares the contents of this register to the next buffer address when linking buffers together. If the contents of this register match the next buffer address the Local DMA operation is aborted.

	7	6	5	4	3	2	1	0
BRNY	A15	A14	A13	A12	A11	A10	A9	A8



## 10.0 Internal Registers (Continued)

### CURRENT PAGE REGISTER (CURR)

This register is used internally by the Buffer Management Logic as a backup register for reception. CURR contains the address of the first buffer to be used for a packet reception and is used to restore DMA pointers in the event of receive errors. This register is initialized to the same value as PSTART and should not be written to again unless the controller is Reset.

	7	6	5	4	3	2	1	0
CURR	A15	A14	A13	A12	A11	A10	A9	A8

### CURRENT LOCAL DMA REGISTER 0,1 (CLDA0,1)

These two registers can be accessed to determine the current local DMA Address.

	7	6	5	4	3	2	1	0
CLDA1	A15	A14	A13	A12	A11	A10	A9	A8

	7	6	5	4	3	2	1	0
CLDA0	A7	A6	A5	A4	A3	A2	A1	A0

## 10.7 REMOTE DMA REGISTERS

### REMOTE START ADDRESS REGISTERS (RSAR0,1)

Remote DMA operations are programmed via the Remote Start Address (RSAR0,1) and Remote Byte Count (RBCR0,1) registers. The Remote Start Address is used to point to the start of the block of data to be transferred and the Remote Byte Count is used to indicate the length of the block (in bytes).

	7	6	5	4	3	2	1	0
RSAR1	A15	A14	A13	A12	A11	A10	A9	A8

	7	6	5	4	3	2	1	0
RSAR0	A7	A6	A5	A4	A3	A2	A1	A0

### 6.4.3.2 REMOTE BYTE COUNT REGISTERS (RBCR0,1)

	7	6	5	4	3	2	1	0
RBCR1	BC15	BC14	BC13	BC12	BC11	BC10	BC9	BC8

	7	6	5	4	3	2	1	0
RBCR0	BC7	BC6	BC5	BC4	BC3	BC2	BC1	BC0

**Note:**

RSAR0 programs the start address bits A0-A7.  
 RSAR1 programs the start address bits A8-A15.  
 Address incremented by two for word transfers, and by one for byte transfers.  
 Byte Count decremented by two for word transfers and by one for byte transfers.  
 RBCR0 programs LSB byte count.  
 RBCR1 programs MSB byte count.

### CURRENT REMOTE DMA ADDRESS (CRDA0, CRDA1)

The Current Remote DMA Registers contain the current address of the Remote DMA. The bit assignment is shown below:

	7	6	5	4	3	2	1	0
CRDA1	A15	A14	A13	A12	A11	A10	A9	A8

	7	6	5	4	3	2	1	0
CRDA0	A7	A6	A5	A4	A3	A2	A1	A0

## 10.8 PHYSICAL ADDRESS REGISTERS (PAR0-PAR5)

The physical address registers are used to compare the destination address of incoming packets for rejecting or accepting packets. Comparisons are performed on a byte-wide basis. The bit assignment shown below relates the sequence in PAR0-PAR5 to the bit sequence of the received packet.

	D7	D6	D5	D4	D3	D2	D1	D0
PAR0	DA7	DA6	DA5	DA4	DA3	DA2	DA1	DA0
PAR1	DA15	DA14	DA13	DA12	DA11	DA10	DA9	DA8
PAR2	DA23	DA22	DA21	DA20	DA19	DA18	DA17	DA16
PAR3	DA31	DA30	DA29	DA28	DA27	DA26	DA25	DA24
PAR4	DA39	DA38	DA37	DA36	DA35	DA34	DA33	DA32
PAR5	DA47	DA46	DA45	DA44	DA43	DA42	DA41	DA40

	Destination Address						Source		
P/S	DA0	DA1	DA2	DA3	.....	DA46	DA47	SA0	...

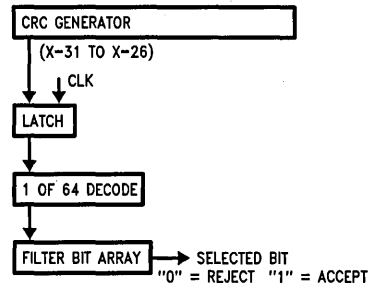
**Note:**

P/S = Preamble, Synch  
 DA0 = Physical/Multicast Bit

## 10.9 MULTICAST ADDRESS REGISTERS (MAR0-MAR7)

The multicast address registers provide filtering of multicast addresses hashed by the CRC logic. All destination addresses are fed through the CRC logic and as the last bit of the destination address enters the CRC, the 6 most significant bits of the CRC generator are latched. These 6 bits are then decoded by a 1 of 64 decode to index a unique filter bit (FB0-63) in the multicast address registers. If the filter bit selected is set, the multicast packet is accepted. The system designer would use a program to determine which filter bits to set in the multicast registers. All multicast filter bits that correspond to multicast address accepted by the node are then set to one. To accept all multicast packets all of the registers are set to all ones.

**Note:** Although the hashing algorithm does not guarantee perfect filtering of multicast address, it will perfectly filter up to 64 multicast addresses if these addresses are chosen to map into unique locations in the multicast filter.



TL/F/8582-62



## 10.0 Internal Registers (Continued)

	D7	D6	D5	D4	D3	D2	D1	D0
MAR0	FB7	FB6	FB5	FB4	FB3	FB2	FB1	FB0
MAR1	FB15	FB14	FB13	FB12	FB11	FB10	FB9	FB8
MAR2	FB23	FB22	FB21	FB20	FB19	FB18	FB17	FB16
MAR3	FB31	FB30	FB29	FB28	FB27	FB26	FB25	FB24
MAR4	FB39	FB38	FB37	FB36	FB35	FB34	FB33	FB32
MAR5	FB47	FB46	FB45	FB44	FB43	FB42	FB41	FB40
MAR6	FB55	FB54	FB53	FB52	FB51	FB50	FB49	FB48
MAR7	FB63	FB62	FB61	FB60	FB59	FB58	FB57	FB56

If address Y is found to hash to the value 32 (20H), then FB32 in MAR4 should be initialized to "1". This will cause the NIC to accept any multicast packet with the address Y.

### NETWORK TALLY COUNTERS

Three 8-bit counters are provided for monitoring the number of CRC errors, Frame Alignment Errors and Missed Packets. The maximum count reached by any counter is 192 (C0H). These registers will be cleared when read by the CPU. The count is recorded in binary in CT0-CT7 of each Tally Register.

#### Frame Alignment Error Tally (CNTR0)

This counter is incremented every time a packet is received with a Frame Alignment Error. The packet must have been recognized by the address recognition logic. The counter is cleared after it is read by the processor.

	7	6	5	4	3	2	1	0
CNTR0	CT7	CT6	CT5	CT4	CT3	CT2	CT1	CT0

#### CRC Error Tally (CNTR1)

This counter is incremented every time a packet is received with a CRC error. The packet must first be recognized by the address recognition logic. The counter is cleared after it is read by the processor.

	7	6	5	4	3	2	1	0
CNTR1	CT7	CT6	CT5	CT4	CT3	CT2	CT1	CT0

#### Frames Lost Tally Register (CNTR2)

This counter is incremented if a packet cannot be received due to lack of buffer resources. In monitor mode, this counter will count the number of packets that pass the address recognition logic.

	7	6	5	4	3	2	1	0
CNTR2	CT7	CT6	CT5	CT4	CT3	CT2	CT1	CT0

### FIFO

This is an eight bit register that allows the CPU to examine the contents of the FIFO after loopback. The FIFO will contain the last 8 data bytes transmitted in the loopback packet. Sequential reads from the FIFO will advance a pointer in the FIFO and allow reading of all 8 bytes.

	7	6	5	4	3	2	1	0
FIFO	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0

**Note:** The FIFO should only be read when the NIC has been programmed in loopback mode.

### NUMBER OF COLLISIONS (NCR)

This register contains the number of collisions a node experiences when attempting to transmit a packet. If no collisions are experienced during a transmission attempt, the COL bit of the TSR will not be set and the contents of NCR will be zero. If there are excessive collisions, the ABT bit in the TSR will be set and the contents of NCR will be zero. The NCR is cleared after the TXP bit in the CR is set.

	7	6	5	4	3	2	1	0
NCR	—	—	—	—	NC3	NC2	NC1	NC0

## 11.0 Initialization Procedures

The NIC must be initialized prior to transmission or reception of packets from the network. Power on reset is applied to the NIC's reset pin. This clears/sets the following bits:

Register	Reset Bits	Set Bits
Command Register (CR)	TXP, STA	RD2, STP
Interrupt Status (ISR)		RST
Interrupt Mask (IMR)	All Bits	
Data Control (DCR)		LAS
Transmit Config. (TCR)	LB1, LB0	

The NIC remains in its reset state until a Start Command is issued. This guarantees that no packets are transmitted or received and that the NIC remains a bus slave until all appropriate internal registers have been programmed. After initialization the STP bit of the command register is reset and packets may be received and transmitted.

### Initialization Sequence

The following initialization procedure is mandatory.

- 1) Program Command Register for Page 0 (Command Register = 21H)
- 2) Initialize Data Configuration Register (DCR)
- 3) Clear Remote Byte Count Registers (RBCR0, RBCR1)
- 4) Initialize Receive Configuration Register (RCR)
- 5) Place the NIC in LOOPBACK mode 1 or 2 (Transmit Configuration Register = 02H or 04H)
- 6) Initialize Receive Buffer Ring: Boundary Pointer (BNDRY), Page Start (PSTART), and Page Stop (PSTOP)
- 7) Clear Interrupt Status Register (ISR) by writing 0FFh to it.
- 8) Initialize Interrupt Mask Register (IMR)
- 9) Program Command Register for page 1 (Command Register = 61H)
  - i) Initialize Physical Address Registers (PAR0-PAR5)
  - ii) Initialize Multicast Address Registers (MAR0-MAR7)
  - iii) Initialize CURRent pointer
- 10) Put NIC in START mode (Command Register = 22H). The local receive DMA is still not active since the NIC is in LOOPBACK.
- 11) Initialize the Transmit Configuration for the intended value. The NIC is now ready for transmission and reception.

## 11.0 Initialization Procedures

(Continued)

Before receiving packets, the user must specify the location of the Receive Buffer Ring. This is programmed in the Page Start and Page Stop Registers. In addition, the Boundary and Current Page Registers must be initialized to the value of the Page Start Register. These registers will be modified during reception of packets.

## 12.0 Loopback Diagnostics

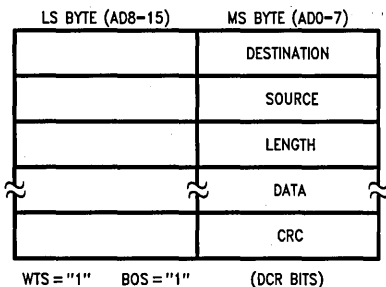
Three forms of local loopback are provided on the NIC. The user has the ability to loopback through the deserializer on the DP8390D NIC, through the DP8391 SNI, and to the coax to check the link through the transceiver circuitry. **Because of the half duplex architecture of the NIC, loopback testing is a special mode of operation with the following restrictions:**

### Restrictions During Loopback

The FIFO is split into two halves, one used for transmission the other for reception. Only 8-bit fields can be fetched from memory so two tests are required for 16-bit systems to verify integrity of the entire data path. During loopback the maximum latency from the assertion of BREQ to BACK is 2.0  $\mu$ s. Systems that wish to use the loopback test yet do not meet this latency can limit the loopback packet to 7 bytes without experiencing underflow. Only the last 8 bytes of the loopback packet are retained in the FIFO. The last 8 bytes can be read through the FIFO register which will advance through the FIFO to allow reading the receive packet sequentially.

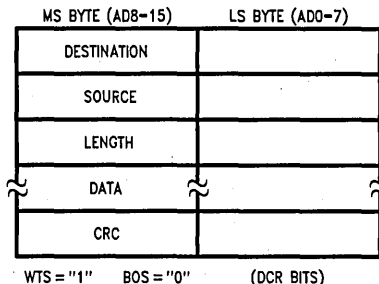
DESTINATION ADDRESS	= (6 bytes) Station Physical Address
SOURCE ADDRESS	
LENGTH	2 bytes
DATA	= 46 to 1500 bytes
CRC	Appended by NIC if CRC = "0" in TCR

When in word-wide mode with Byte Order Select set, the loopback packet must be assembled in the even byte locations as shown below. (The loopback only operates with byte wide transfers.)



TL/F/6582-15

When in word-wide mode with Byte Order Select low, the following format must be used for the loopback packet.



TL/F/6582-16

**Note:** When using loopback in word mode 2n bytes must be programmed in TBCRO, 1. Where n = actual number of bytes assembled in even or odd location.

To initiate a loopback the user first assembles the loopback packet then selects the type of loopback using the Transmit Configuration register bits LB0, LB1. The transmit configuration register must also be set to enable or disable CRC generation during transmission. The user then issues a normal transmit command to send the packet. During loopback the receiver checks for an address match and if CRC bit in the TCR is set, the receiver will also check the CRC. The last 8 bytes of the loopback packet are buffered and can be read out of the FIFO using the FIFO read port.

### Loopback Modes

**MODE 1: Loopback Through the Controller (LB1 = 0, LB0 = 1).**

If the loopback is through the NIC then the serializer is simply linked to the deserializer and the receive clock is derived from the transmit clock.

**MODE 2: Loopback Through the SNI (LB1 = 1, LB0 = 0).**

If the loopback is to be performed through the SNI, the NIC provides a control (LPBK) that forces the SNI to loopback all signals.

**MODE 3: Loopback to Coax (LB1 = 1, LB0 = 1).**

Packets can be transmitted to the coax in loopback mode to check all of the transmit and receive paths and the coax itself.

**Note:** In MODE 1, CRS and COL lines are not indicated in any status register, but the NIC will still defer if these lines are active. In MODE 2, COL is masked and in MODE 3 CRS and COL are not masked. It is not possible to go directly between the loopback modes, it is necessary to return to normal operation (00H) when changing modes.

### Reading the Loopback Packet

The last eight bytes of a received packet can be examined by 8 consecutive reads of the FIFO register. The FIFO pointer is incremented after the rising edge of the CPU's read strobe by internally synchronizing and advancing the pointer. This may take up to four bus clock cycles, if the pointer has not been incremented by the time the CPU reads the FIFO register again, the NIC will insert wait states

**Note:** The FIFO may only be read during Loopback. Reading the FIFO at any other time will cause the NIC to malfunction.

## 12.0 Loopback Diagnostics (Continued)

### Alignment of the Received Packet in the FIFO

Reception of the packet in the FIFO begins at location zero, after the FIFO pointer reaches the last location in the FIFO, the pointer wraps to the top of the FIFO overwriting the previously received data. This process continues until the last byte is received. The NIC then appends the received byte count in the next two locations of the FIFO. The contents of the Upper Byte Count are also copied to the next FIFO location. The number of bytes used in the loopback packet determines the alignment of the packet in the FIFO. The alignment for a 64-byte packet is shown below.

FIFO LOCATION	FIFO CONTENTS		
0	LOWER BYTE COUNT	→	First Byte Read
1	UPPER BYTE COUNT	→	Second Byte Read
2	UPPER BYTE COUNT		•
3	LAST BYTE		•
4	CRC1		•
5	CRC2		•
6	CRC3		•
7	CRC4	→	Last Byte Read

For the following alignment in the FIFO the packet length should be  $(N \times 8) + 5$  Bytes. Note that if the CRC bit in the TCR is set, CRC will not be appended by the transmitter. If the CRC is appended by the transmitter, the last four bytes, bytes N-3 to N, correspond to the CRC.

FIFO LOCATION	FIFO CONTENTS		
0	BYTE N-4	→	First Byte Read
1	BYTE N-3 (CRC1)	AR	Second Byte Read
2	BYTE N-2 (CRC2)		•
3	BYTE N-1 (CRC3)		•
4	BYTE N (CRC4)		•
5	LOWER BYTE COUNT		•
6	UPPER BYTE COUNT	→	Last Byte Read
7	UPPER BYTE COUNT		

### LOOPBACK TESTS

Loopback capabilities are provided to allow certain tests to be performed to validate operation of the DP8390D NIC prior to transmitting and receiving packets on a live network. Typically these tests may be performed during power up of a node. The diagnostic provides support to verify the following:

- 1) Verify integrity of data path. Received data is checked against transmitted data.
- 2) Verify CRC logic's capability to generate good CRC on transmit, verify CRC on receive (good or bad CRC).
- 3) Verify that the Address Recognition Logic can
  - a) Recognize address match packets
  - b) Reject packets that fail to match an address

### LOOPBACK OPERATION IN THE NIC

Loopback is a modified form of transmission using only half of the FIFO. This places certain restrictions on the use of loopback testing. When loopback mode is selected in the TCR, the FIFO is split. A packet should be assembled in memory with programming of TPSR and TBCR0, TBCR1 registers. When the transmit command is issued the following operations occur:

#### Transmitter Actions

- 1) Data is transferred from memory by the DMA until the FIFO is filled. For each transfer TBCR0 and TBCR1 are decremented. (Subsequent burst transfers are initiated when the number of bytes in the FIFO drops below the programmed threshold.)
- 2) The NIC generates 56 bits of preamble followed by an 8-bit synch pattern.
- 3) Data transferred from FIFO to serializer.
- 4) If CRC=1 in TCR, no CRC calculated by NIC, the last byte transmitted is the last byte from the FIFO (Allows software CRC to be appended). If CRC=0, NIC calculates and appends four bytes of CRC.
- 5) At end of Transmission PTX bit set in ISR.

#### Receiver Actions

- 1) Wait for synch, all preamble stripped.
- 2) Store packet in FIFO, increment receive byte count for each incoming byte.
- 3) If CRC=0 in TCR, receiver checks incoming packet for CRC errors. If CRC=1 in TCR, receiver does not check CRC errors, CRC error bit always set in RSR (for address matching packets).
- 4) At end of receive, receive byte count written into FIFO, receive status register is updated. The PRX bit is typically set in the RSR even if the address does not match. If CRC errors are forced, the packet must match the address filters in order for the CRC error bit in the RS to be set.

### EXAMPLES

The following examples show what results can be expected from a properly operating NIC during loopback. The restrictions and results of each type of loopback are listed for reference. The loopback tests are divided into two sets of tests. One to verify the data path, CRC generation and byte count through all three paths. The second set of tests uses internal loopback to verify the receiver's CRC checking and address recognition. For all of the tests the DCR was programmed to 40h.

PATH	TCR	RCR	TSR	RSR	ISR
NIC Internal	02	00	53(1)	02(2)	02(3)

**Note 1:** Since carrier sense and collision detect inputs are blocked during internal loopback, carrier and CD heartbeat are not seen and the CRS and CDH bits are set.

**Note 2:** CRC errors are always indicated by receiver if CRC is appended by the transmitter.

**Note 3:** Only the PTX bit in the ISR is set, the PRX bit is only set if status is written to memory. In loopback this action does not occur and the PRX bit remains 0 for all loopback modes.

**Note 4:** All values are hex.

## 12.0 Loopback Diagnostics (Continued)

PATH	TCR	RCR	TSR	RSR	ISR
NIC External	04	00	43(1)	02	02

**Note 1:** CDH is set, CRS is not set since it is generated by the external encoder/decoder.

PATH	TCR	RCR	TSR	RSR	ISR
NIC External	06	00	03(1)	02	02(2)

**Note 1:** CDH and CRS should not be set. The TSR however, could also contain 01H,03H,07H and a variety of other values depending on whether collisions were encountered or the packet was deferred.

**Note 2:** Will contain 08H if packet is not transmittable.

**Note 3:** During external loopback the NIC is now exposed to network traffic, it is therefore possible for the contents of both the Receive portion of the FIFO and the RSR to be corrupted by any other packet on the network. Thus in a live network the contents of the FIFO and RSR should not be depended on. The NIC will still abide by the standard CSMA/CD protocol in external loopback mode. (i.e. The network will not be disturbed by the loopback packet).

**Note 4:** All values are hex.

### CRC AND ADDRESS RECOGNITION

The next three tests exercise the address recognition logic and CRC. These tests should be performed using internal loopback only so that the NIC is isolated from interference from the network. These tests also require the capability to generate CRC in software.

The address recognition logic cannot be directly tested. The CRC and FAE bits in the RSR are only set if the address of the packet matches the address filters. If errors are expected to be set and they are not set, the packet has been rejected on the basis of an address mismatch. The following sequence of packets will test the address recognition logic. The DCR should be set to 40H, the TCR should be set to 03H with a software generated CRC.

Packet Contents			Results
Test	Address	CRC	RSR
Test A	Matching	Good	01(1)
Test B	Matching	Bad	02(2)
Test C	Non-Matching	Bad	01

**Note 1:** Status will read 21H if multicast address used.

**Note 2:** Status will read 22H if multicast address used.

**Note 3:** In test A, the RSR is set up. In test B the address is found to match since the CRC is flagged as bad. Test C proves that the address recognition logic can distinguish a bad address and does not notify the RSR of the bad CRC. The receiving CRC is proven to work in test A and test B.

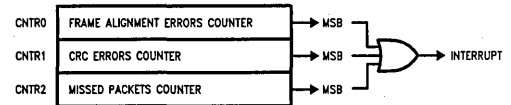
**Note 4:** All values are hex.

### NETWORK MANAGEMENT FUNCTIONS

Network management capabilities are required for maintenance and planning of a local area network. The NIC supports the minimum requirement for network management in hardware, the remaining requirements can be met with software counts. There are three events that software alone can not track during reception of packets: CRC errors, Frame Alignment errors, and missed packets.

Since errored packets can be rejected, the status associated with these packets is lost unless the CPU can access the Receive Status Register before the next packet arrives. In situations where another packet arrives very quickly, the CPU may have no opportunity to do this. The NIC counts the number of packets with CRC errors and Frame Alignment errors. 8-bit counters have been selected to reduce overhead. The counters will generate interrupts whenever their MSBs are set so that a software routine can accumulate the network statistics and reset the counters before overflow occurs. The counters are sticky so that when they reach a count of 192 (COH) counting is halted. An additional counter is provided to count the number of packets NIC misses due to buffer overflow or being offline.

The structure of the counters is shown below:



TL/F/8582-63

Additional information required for network management is available in the Receive and Transmit Status Registers. Transmit status is available after each transmission for information regarding events during transmission.

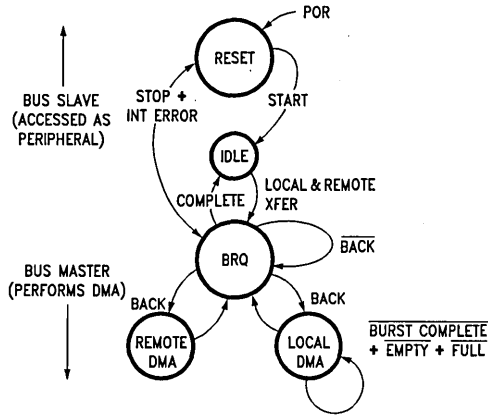
Typically, the following statistics might be gathered in software:

- Traffic:
  - Frames Sent OK
  - Frames Received OK
  - Multicast Frames Received
  - Packets Lost Due to Lack of Resources
  - Retries/Packet
- Errors:
  - CRC Errors
  - Alignment Errors
  - Excessive Collisions
  - Packet with Length Errors
  - Heartbeat Failure

### 13.0 Bus Arbitration and Timing

The NIC operates in three possible modes:

- BUS MASTER (WHILE PERFORMING DMA)
- BUS SLAVE (WHILE BEING ACCESSED BY CPU)
- IDLE



TL/F/8582-64

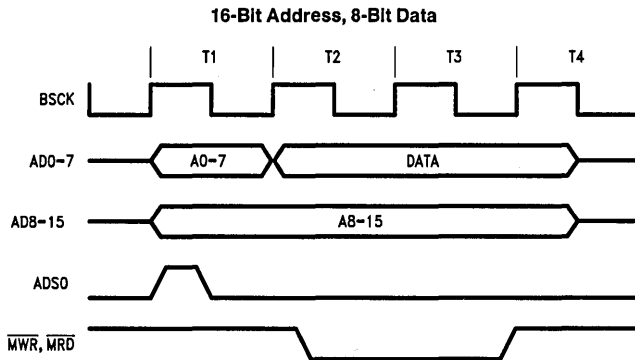
Upon power-up the NIC is in an indeterminate state. After receiving a Hardware Reset the NIC comes up as a slave in the Reset State. The receiver and transmitter are both disabled in this state. The reset state can be reentered under three conditions, soft reset (Stop Command), hard reset (RESET input) or an error that shuts down the receiver or transmitter (FIFO underflow or overflow). After initialization of registers, the NIC is issued a Start command and the NIC enters Idle state. Until the DMA is required the NIC remains in an idle state. The idle state is exited by a request from the FIFO in the case of receive or transmit, or from the Remote/DMA in the case of Remote DMA operation. After acquiring the bus in a BREQ/BACK handshake the Remote or Local DMA transfer is completed and the NIC reenters the idle state.

#### DMA TRANSFERS TIMING

The DMA can be programmed for the following types of transfers:

- 16-Bit Address, 8-bit Data Transfer
- 16-Bit Address, 16-bit Data Transfer
- 32-Bit Address, 8-bit Data Transfer
- 32-Bit Address, 16-bit Data Transfer

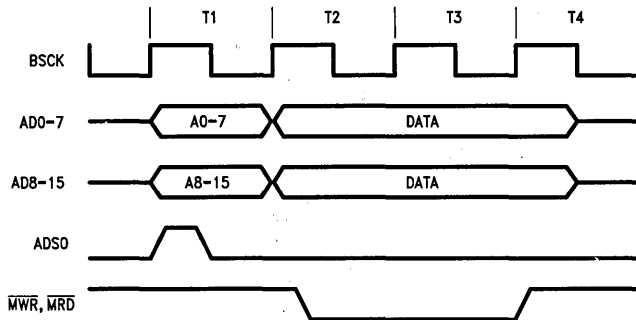
All DMA transfers use BSCK for timing. 16-Bit Address modes require 4 BSCK cycles as shown below:



TL/F/8582-65

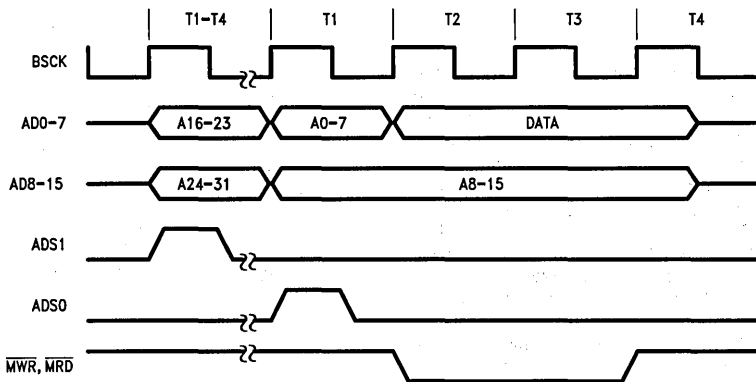
### 13.0 Bus Arbitration and Timing (Continued)

16-Bit Address, 16-Bit Data



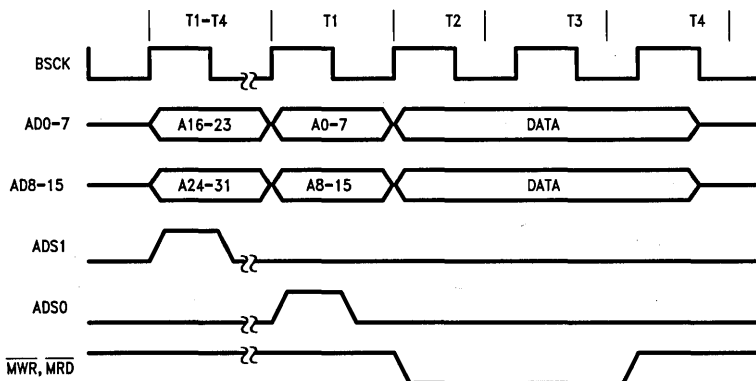
TL/F/8582-66

32-Bit Address, 8-Bit Data



TL/F/8582-67

32-Bit Address, 16-Bit Data



TL/F/8582-68

Note: In 32-bit address mode, ADS1 is at TRI-STATE after the first T1-T4 states; thus, a 4.7k pull-down resistor is required for 32-bit address mode.

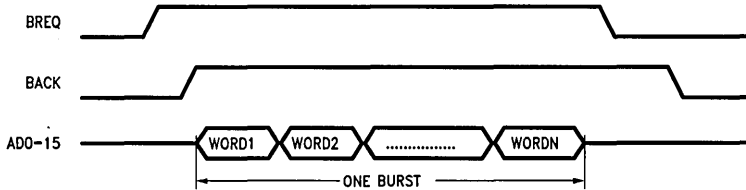
### 13.0 Bus Arbitration and Timing (Continued)

When in 32-bit mode four additional BSCK cycles are required per burst. The first bus cycle (T1'-T4') of each burst is used to output the upper 16-bit addresses. This 16-bit address is programmed in RSAR0 and RSAR1 and points to a 64k page of system memory. All transmitted or received packets are constrained to reside within this 64k page.

#### FIFO BURST CONTROL

All Local DMA transfers are burst transfers, once the DMA requests the bus and the bus is acknowledged, the DMA will

transfer an exact burst of bytes programmed in the Data Configuration Register (DCR) then relinquish the bus. If there are remaining bytes in the FIFO the next burst will not be initiated until the FIFO threshold is exceeded. If BACK is removed during the transfer, the burst transfer will be aborted. **(DROPPING BACK DURING A DMA CYCLE IS NOT RECOMMENDED.)**



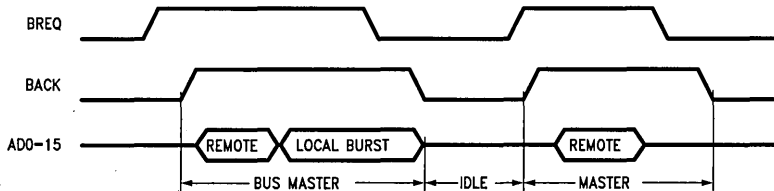
TL/F/8582-69

where N = 1, 2, 4, or 6 Words or N = 2, 4, 8, or 12 Bytes when in byte mode

#### INTERLEAVED LOCAL OPERATION

If a remote DMA transfer is initiated or in progress when a packet is being received or transmitted, the Remote DMA transfer will be interrupted for higher priority Local DMA

transfers. When the Local DMA transfer is completed the Remote DMA will re-arbitrate for the bus and continue its transfers. This is illustrated below:



TL/F/8582-70

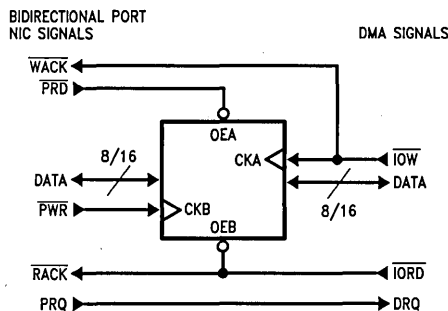
Note that if the FIFO requires service while a remote DMA is in progress, BREQ is not dropped and the Local DMA burst is appended to the Remote Transfer. When switching from a local transfer to a remote transfer, however, BREQ is dropped and raised again. This allows the CPU or other devices to fairly contend for the bus.

This transfer is arbitrated on a byte by byte basis versus the burst transfer used for Local DMA transfers. This bidirectional port is also read/written by the host. All transfers through this port are asynchronous. At any one time transfers are limited to one direction, either from the port to local buffer memory (Remote Write) or from local buffer memory to the port (Remote Read).

#### REMOTE DMA-BIDIRECTIONAL PORT CONTROL

The Remote DMA transfers data between the local buffer memory and a bidirectional port (memory to I/O transfer).

#### Bus Handshake Signals for Remote DMA Transfers



TL/F/8582-71

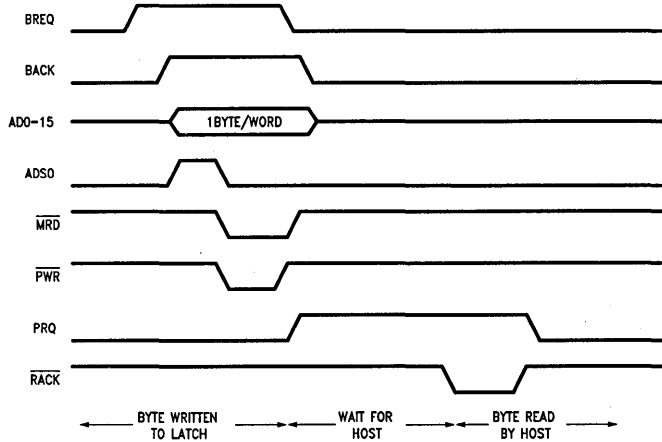
### 13.0 Bus Arbitration and Timing (Continued)

#### REMOTE READ TIMING

- 1) The DMA reads byte/word from local buffer memory and writes byte/word into latch, increments the DMA address and decrements the byte count (RBCR0,1).
- 2) A Request Line (PRQ) is asserted to inform the system that a byte is available.
- 3) The system reads the port, the read strobe ( $\overline{\text{RACK}}$ ) is used as an acknowledge by the Remote DMA and it goes back to step 1.

Steps 1-3 are repeated until the remote DMA is complete.

Note that in order for the Remote DMA to transfer a byte from memory to the latch, it must arbitrate access to the local bus via a BREQ, BACK handshake. After each byte or word is transferred to the latch, BREQ is dropped. If a Local DMA is in progress, the Remote DMA is held off until the local DMA is complete.



TL/F/8582-72

#### REMOTE WRITE TIMING

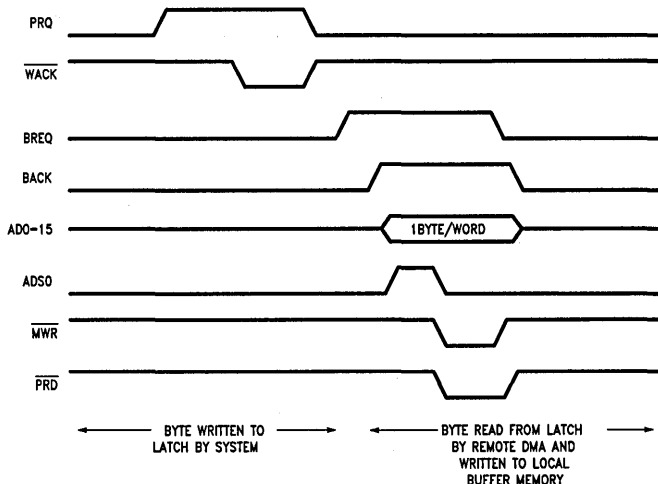
A Remote Write operation transfers data from the I/O port to the local buffer RAM. The NIC initiates a transfer by requesting a byte/word via the PRQ. The system transfers a byte/word to the latch via  $\overline{\text{IOW}}$ , this write strobe is detected by the NIC and PRQ is removed. By removing the PRQ, the Remote DMA holds off further transfers into the latch until the current byte/word has been transferred from the latch, PRQ is reasserted and the next transfer can begin.

- 1) NIC asserts PRQ. System writes byte/word into latch. NIC removes PRQ.

- 2) Remote DMA reads contents of port and writes byte/word to local buffer memory, increments address and decrements byte count (RBCR0,1).

- 3) Go back to step 1.

Steps 1-3 are repeated until the remote DMA is complete.



TL/F/8582-73

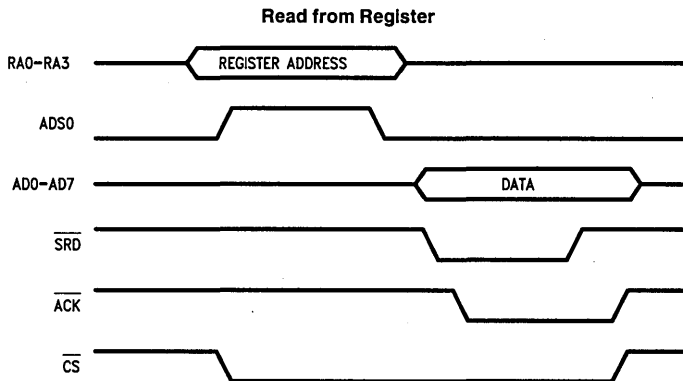
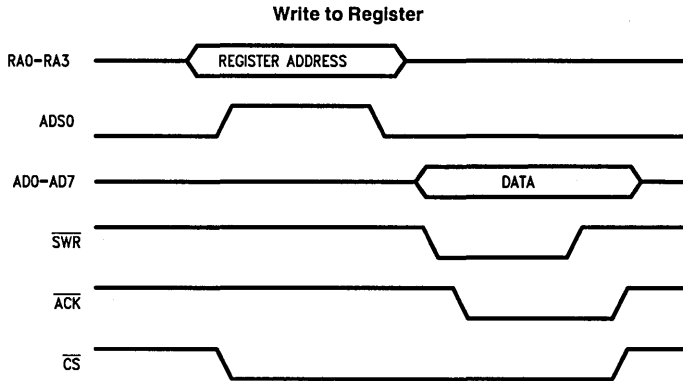


### 13.0 Bus Arbitration and Timing (Continued)

#### SLAVE MODE TIMING

When  $\overline{CS}$  is low, the NIC becomes a bus slave. The CPU can then read or write any internal registers. All register access is byte wide. The timing for register access is shown below. The host CPU accesses internal registers with four address lines, RA0-RA3,  $\overline{SRD}$  and  $\overline{SWR}$  strobes.

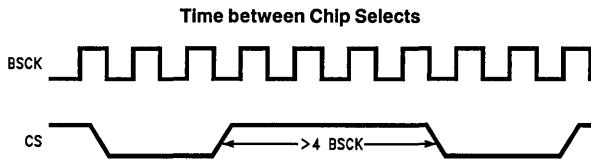
ADS0 is used to latch the address when interfacing to a multiplexed, address data bus. Since the NIC may be a local bus master when the host CPU attempts to read or write to the controller, an  $\overline{ACK}$  line is used to hold off the CPU until the NIC leaves master mode. Some number of BSCK cycles is also required to allow the NIC to synchronize to the read or write cycle.



#### TIME BETWEEN CHIP SELECTS

The NIC requires that successive chip selects be no closer than 4 bus clocks (BSCK) together, below. If the condition is violated, the NIC may glitch/ $\overline{ACK}$ . CPUs that operate from pipelined instructions (i.e. 386) or have a cache (i.e.

486) can execute consecutive I/O cycles very quickly. The solution is to delay the execution of consecutive I/O cycles by either breaking the pipeline or forcing the CPU to access outside its cache.



## 14.0 Preliminary Electrical Characteristics

### Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage ( $V_{CC}$ )	-0.5V to +7.0V
DC Input Voltage ( $V_{IN}$ )	-0.5V to $V_{CC} + 0.5V$
DC Output Voltage ( $V_{OUT}$ )	-0.5V to $V_{CC} + 0.5V$
Storage Temperature Range ( $T_{STG}$ )	-65°C to +150°C
Power Dissipation (PD)	500 mW
Lead Temp. (TL) (Soldering, 10 sec.)	260°C
ESD rating ( $R_{ZAP} = 1.5k, C_{ZAP} = 120$ pF)	1600V

### Preliminary DC Specifications $T_A = 0^\circ\text{C}$ to $70^\circ\text{C}$ , $V_{CC} = 5V \pm 5\%$ , unless otherwise specified

Symbol	Parameter	Conditions	Min	Max	Units
$V_{OH}$	Minimum High Level Output Voltage (Notes 1, 4)	$I_{OH} = -20 \mu\text{A}$	$V_{CC} - 0.1$		V
		$I_{OH} = -2.0 \text{ mA}$	3.5		V
$V_{OL}$	Minimum Low Level Output Voltage (Notes 1, 4)	$I_{OL} = 20 \mu\text{A}$		0.1	V
		$I_{OL} = 2.0 \text{ mA}$		0.4	V
$V_{IH}$	Minimum High Level Input Voltage (Note 2)		2.0		V
$V_{IH2}$	Minimum High Level Input Voltage for RACK, WACK (Note 2)		2.7		V
$V_{IL}$	Minimum Low Level Input Voltage (Note 2)			0.8	V
$V_{IL2}$	Minimum Low Level Input Voltage For RACK, WACK (Note 2)			0.6	V
$I_{IN}$	Input Current	$V_I = V_{CC}$ or GND	-1.0	+1.0	$\mu\text{A}$
$I_{OZ}$	Maximum TRI-STATE Output Leakage Current	$V_{OUT} = V_{CC}$ or GND	-10	+10	$\mu\text{A}$
$I_{CC}$	Average Supply Current (Note 3)	TXCK = 10 MHz RXCK = 10 MHz BSCK = 20 MHz $I_{OUT} = 0 \mu\text{A}$ $V_{IN} = V_{CC}$ or GND		40	mA

**Note 1:** These levels are tested dynamically using a limited amount of functional test patterns, please refer to AC Test Load.

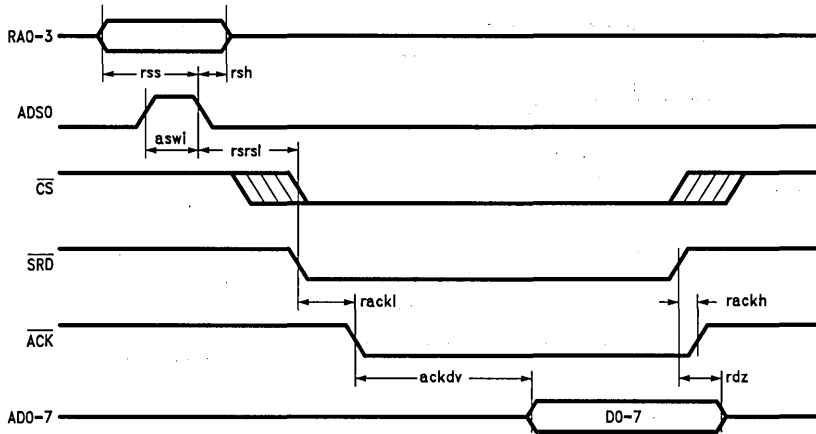
**Note 2:** Limited functional test patterns are performed at these input levels. The majority of functional tests are performed at levels of 0V and 3V.

**Note 3:** This is measured with a 0.1  $\mu\text{F}$  bypass capacitor between  $V_{CC}$  and GND.

**Note 4:** The low drive CMOS compatible  $V_{OH}$  and  $V_{OL}$  limits are not tested directly. Detailed device characterization validates that this specification can be guaranteed by testing the high drive TTL compatible  $V_{OL}$  and  $V_{OH}$  specification.

## 15.0 Switching Characteristics AC Specs DP8390D Note: All Timing is Preliminary

### Register Read (Latched Using ADS0)



TL/F/8582-76

Symbol	Parameter	Min	Max	Units
rss	Register Select Setup to ADS0 Low	10		ns
rsh	Register Select Hold from ADS0 Low	13		ns
aswi	Address Strobe Width In	15		ns
ackdv	Acknowledge Low to Data Valid		55	ns
rdz	Read Strobe to Data TRI-STATE	15	70	ns
rackl	Read Strobe to $\overline{ACK}$ Low (Notes 1, 3)		$n * bcyc + 30$	ns
rackh	Read Strobe to $\overline{ACK}$ High		30	ns
rsrsl	Register Select to Slave Read Low, Latched RS0-3 (Note 2)	10		ns

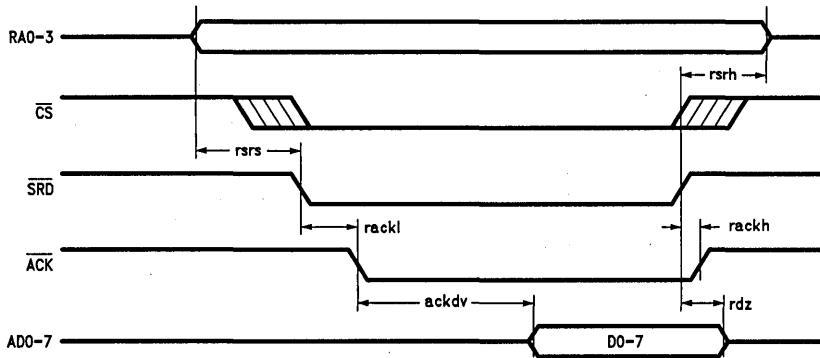
**Note 1:**  $\overline{ACK}$  is not generated until  $\overline{CS}$  and  $\overline{SRD}$  are low and the NIC has synchronized to the register access. The NIC will insert an integral number of Bus Clock cycles until it is synchronized. In Dual Bus systems additional cycles will be used for a local or remote DMA to complete. Wait states must be issued to the CPU until  $\overline{ACK}$  is asserted low.

**Note 2:**  $\overline{CS}$  may be asserted before or after  $\overline{SRD}$ . If  $\overline{CS}$  is asserted after  $\overline{SRD}$ , rackl is referenced from falling edge of  $\overline{CS}$ .  $\overline{CS}$  can be de-asserted concurrently with  $\overline{SRD}$  or after  $\overline{SRD}$  is de-asserted.

**Note 3:** These limits include the RC delay inherent in our test method. These signals typically turn off within 15 ns, enabling other devices to drive these lines with no contention.

## 15.0 Switching Characteristics (Continued)

Register Read (Non Latched, ADS0 = 1)



TL/F/8582-77

Symbol	Parameter	Min	Max	Units
rsrs	Register Select to Read Setup (Notes 1, 3)	10		ns
rsrh	Register Select Hold from Read	0		ns
ackdv	ACK Low to Valid Data		55	ns
rdz	Read Strobe to Data TRI-STATE (Note 2)	15	70	ns
rackl	Read Strobe to ACK Low (Note 3)		$n \cdot \text{bcyc} + 30$	ns
rackh	Read Strobe to ACK High		30	ns

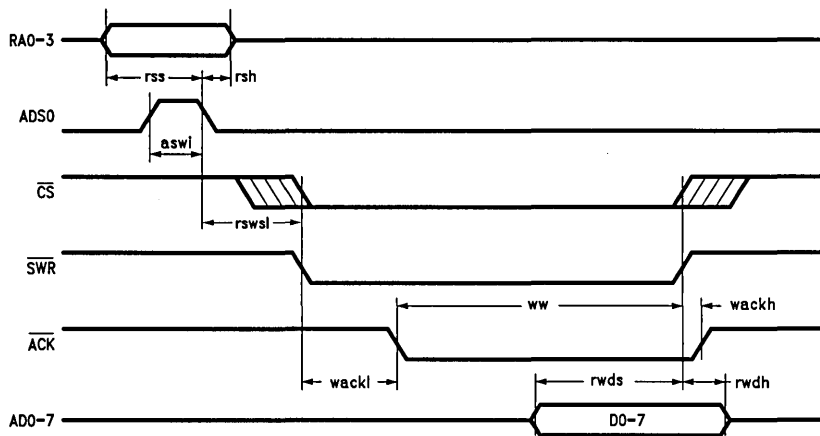
**Note 1:** rsrs includes flow-through time of latch.

**Note 2:** These limits include the RC delay inherent in our test method. These signals typically turn off within 15 ns enabling other devices to drive these lines with no contention.

**Note 3:** CS may be asserted before or after RA0-3, and SRD, since address decode begins when ACK is asserted. If CS is asserted after RA0-3, and SRD, rack1 is referenced from falling edge of CS.

## 15.0 Switching Characteristics (Continued)

Register Write (Latched Using ADS0)



TL/F/8582-78

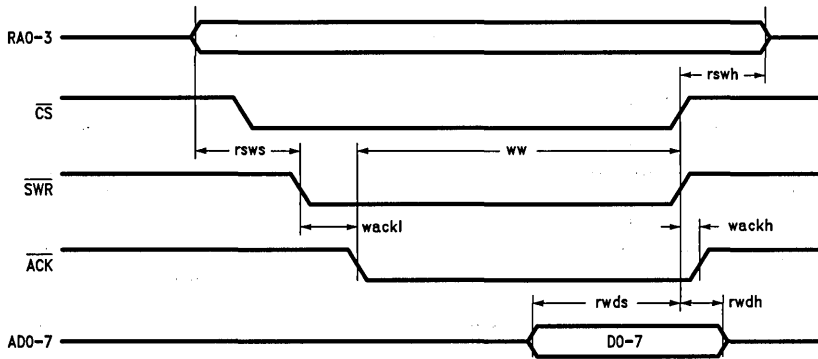
Symbol	Parameter	Min	Max	Units
rss	Register Select Setup to ADS0 Low	10		ns
rsh	Register Select Hold from ADS0 Low	17		ns
aswi	Address Strobe Width In	15		ns
rwds	Register Write Data Setup	20		ns
rwdh	Register Write Data Hold	21		ns
ww	Write Strobe Width from $\overline{ACK}$	50		ns
wackh	Write Strobe High to $\overline{ACK}$ High		30	ns
wackl	Write Low to $\overline{ACK}$ Low (Notes 1, 2)		$n \cdot \text{bcyc} + 30$	ns
rsysl	Register Select to Write Strobe Low	10		ns

**Note 1:**  $\overline{ACK}$  is not generated until  $\overline{CS}$  and  $\overline{SWR}$  are low and the NIC has synchronized to the register access. In Dual Bus Systems additional cycles will be used for a local DMA or Remote DMA to complete.

**Note 2:**  $\overline{CS}$  may be asserted before or after  $\overline{SWR}$ . If  $\overline{CS}$  is asserted after  $\overline{SWR}$ , wackl is referenced from falling edge of  $\overline{CS}$ .

## 15.0 Switching Characteristics (Continued)

Register Write (Non Latched, ADS0 = 1)



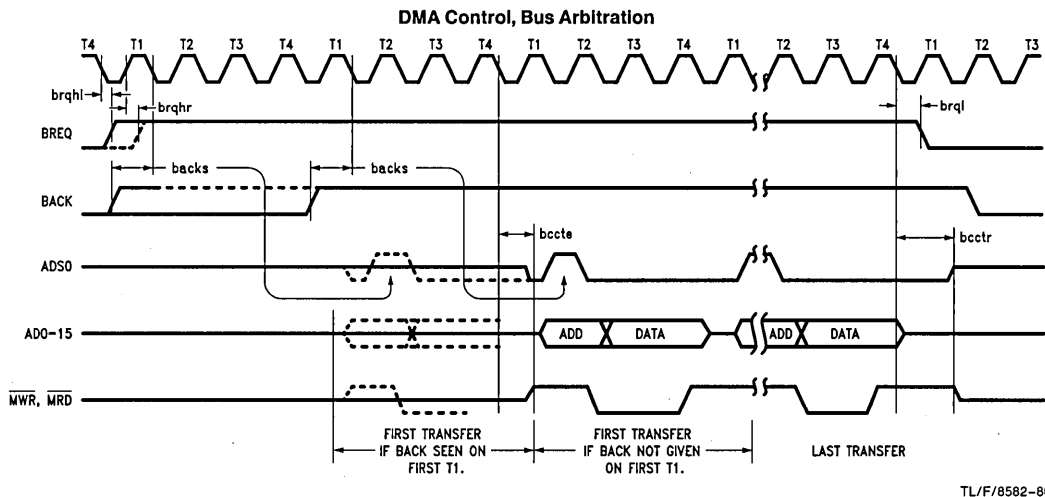
TL/F/8582-79

Symbol	Parameter	Min	Max	Units
rsws	Register Select to Write Setup (Note 1)	15		ns
rswl	Register Select Hold from Write	0		ns
rlds	Register Write Data Setup	20		ns
rldh	Register Write Data Hold	21		ns
wackl	Write Low to $\overline{ACK}$ Low (Note 2)		$n \cdot \text{bcyc} + 30$	ns
wackh	Write High to $\overline{ACK}$ High		30	ns
ww	Write Width from $\overline{ACK}$	50		ns

**Note 1:** Assumes ADS0 is high when RA0-3 changing.

**Note 2:**  $\overline{ACK}$  is not generated until  $\overline{CS}$  and  $\overline{SWR}$  are low and the NIC has synchronized to the register access. In Dual Bus systems additional cycles will be used for a local DMA or remote DMA to complete.

## 15.0 Switching Characteristics (Continued)



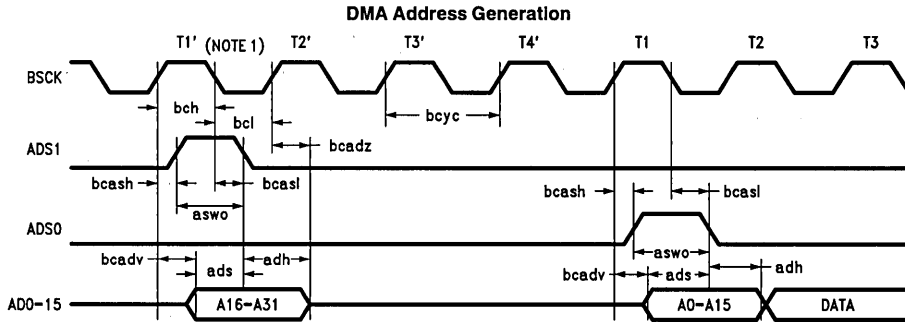
Symbol	Parameter	Min	Max	Units
brqhl	Bus Clock to Bus Request High for Local DMA		43	ns
brqhr	Bus Clock to Bus Request High for Remote DMA		38	ns
brql	Bus Request Low from Bus Clock		55	ns
backs	Acknowledge Setup to Bus Clock (Note 1)	2		ns
bccte	Bus Clock to Control Enable		60	ns
bcctr	Bus Clock to Control Release (Notes 2, 3)		70	ns

**Note 1:** BACK must be setup before T1 after BREQ is asserted. Missed setup will slip the beginning of the DMA by four bus clocks. The Bus Latency will influence the allowable FIFO threshold and transfer mode (empty/fill vs exact burst transfer).

**Note 2:** During remote DMA transfers only, a single bus transfer is performed. During local DMA operations burst mode transfers are performed.

**Note 3:** These limits include the RC delay inherent in our test method. These signals typically turn off within 15 ns enabling other devices to drive these lines with no contention.

## 15.0 Switching Characteristics (Continued)



TL/F/8582-81

Symbol	Parameter	Min	Max	Units
bcyc	Bus Clock Cycle Time (Note 2)	50	1000	ns
bch	Bus Clock High Time	22.5		ns
bcl	Bus Clock Low Time	22.5		ns
bcash	Bus Clock to Address Strobe High		34	ns
bcasl	Bus Clock to Address Strobe Low		44	ns
aswo	Address Strobe Width Out	bch		ns
bcadv	Bus Clock to Address Valid		45	ns
bcadz	Bus Clock to Address TRI-STATE (Note 3)	15	55	ns
ads	Address Setup to ADS0/1 Low	bch - 15		ns
adh	Address Hold from ADS0/1 Low	bcl - 5		ns

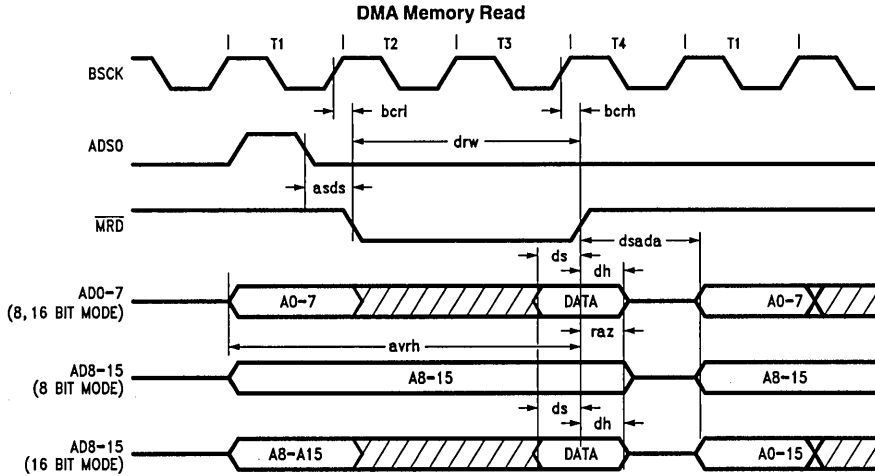
**Note 1:** Cycles T1', T2', T3', T4' are only issued for the first transfer in a burst when 32-bit mode has been selected.

**Note 2:** The rate of bus clock must be high enough to support transfers to/from the FIFO at a rate greater than the serial network transfers from/to the FIFO.

**Note 3:** These limits include the RC delay inherent in our test method. These signals typically turn off within 15 ns, enabling other devices to drive these lines with no contention.



### 15.0 Switching Characteristics (Continued)



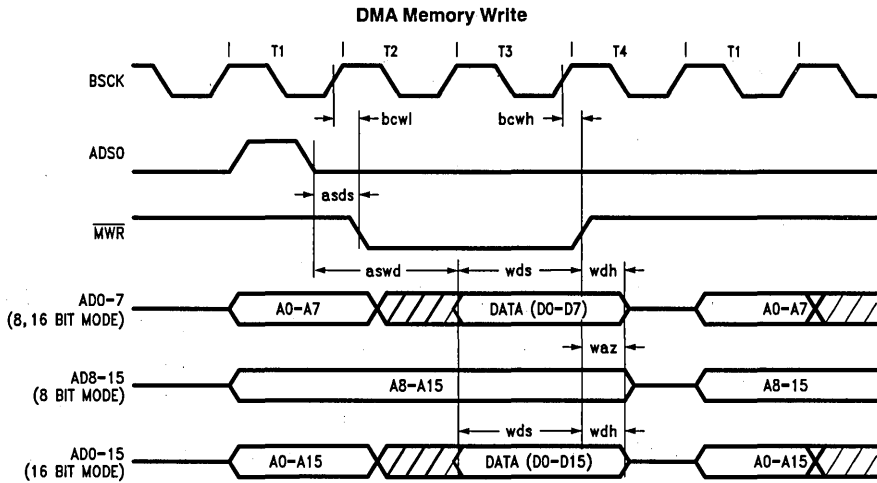
TL/F/8582-82

Symbol	Parameter	Min	Max	Units
bcrh	Bus Clock to Read Strobe Low		43	ns
bcrh	Bus Clock to Read Strobe High		40	ns
ds	Data Setup to Read Strobe High	25		ns
dh	Data Hold from Read Strobe High	0		ns
drw	DMA Read Strobe Width Out	$2 \cdot \text{bcyc} - 15$		ns
raz	Memory Read High to Address TRI-STATE (Notes 1, 2)		bch + 40	ns
asds	Address Strobe to Data Strobe		bcl + 10	ns
dsada	Data Strobe to Address Active	$\text{bcyc} - 10$		ns
avrh	Address Valid to Read Strobe High	$3 \cdot \text{bcyc} - 15$		ns

**Note 1:** During a burst A8-A15 are not TRI-STATE if byte wide transfers are selected. On the last transfer A8-A15 are TRI-STATE as shown above.

**Note 2:** These limits include the RC delay inherent in our test method. These signals typically turn off within bch + 15 ns, enabling other devices to drive these lines with no contention.

### 15.0 Switching Characteristics (Continued)



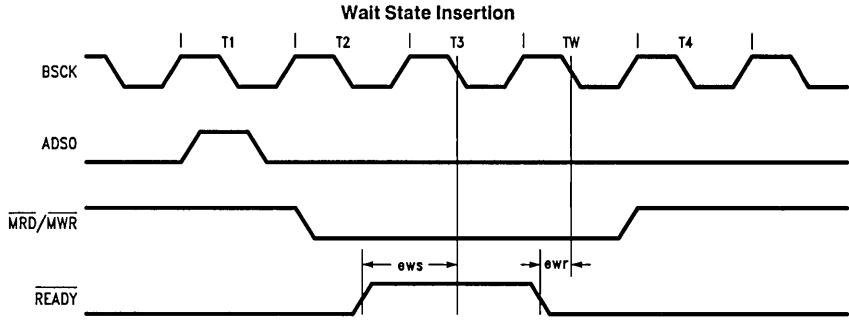
TL/F/8582-83

Symbol	Parameter	Min	Max	Units
bcwl	Bus Clock to Write Strobe Low		40	ns
bcwh	Bus Clock to Write Strobe High		40	ns
wds	Data Setup to $\overline{WR}$ High	$2 * bcyc - 30$		ns
wdh	Data Hold from $\overline{WR}$ Low	$bch + 7$		ns
waz	Write Strobe to Address TRI-STATE (Notes 1, 2)		$bch + 40$	ns
asds	Address Strobe to Data Strobe		$bcl + 10$	ns
aswd	Address Strobe to Write Data Valid		$bcl + 30$	ns

**Note 1:** When using byte mode transfers A8-A15 are only TRI-STATE on the last transfer, waz timing is only valid for last transfer in a burst.

**Note 2:** These limits include the RC delay inherent in our test method. These signals typically turn off within  $bch + 15$  ns, enabling other devices to drive these lines with no contention.

### 15.0 Switching Characteristics (Continued)



TL/F/8582-45

Symbol	Parameter	Min	Max	Units
e <sub>ws</sub>	External Wait Setup to T <sub>3</sub> ↓ Clock (Note 1)	10		ns
e <sub>wr</sub>	External Wait Release Time (Note 1)	15		ns

**Note 1:** The addition of wait states affects the count of deserialized bytes and is limited to a number of bus clock cycles depending on the bus clock and network rates. The allowable wait states are found in the table below. (Assumes 10 Mbit/sec data rate.)

BSCCK (MHz)	# of Wait States	
	Byte Transfer	Word Transfer
8	0	1
10	0	1
12	1	2
14	1	2
16	1	3
18	2	3
20	2	4

Table assumes 10 MHz network clock.

The number of allowable wait states in byte mode can be calculated using:

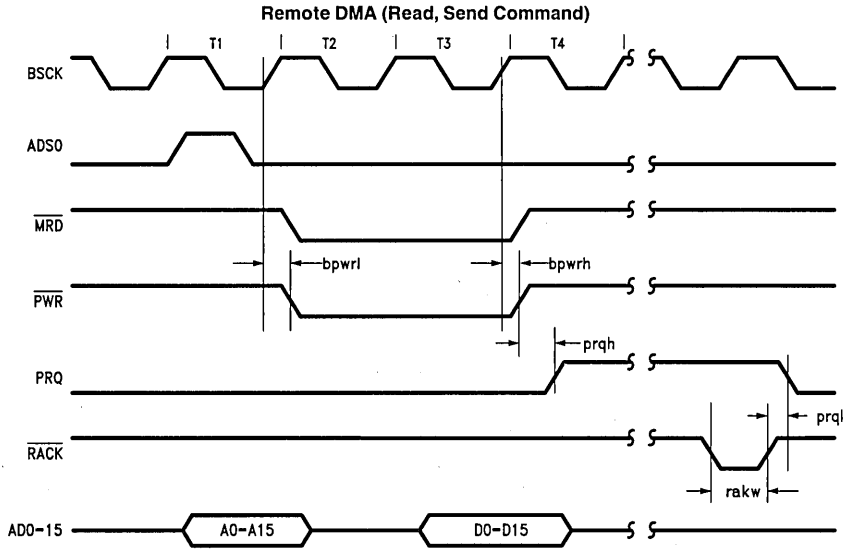
$$\#W_{(\text{byte mode})} = \left( \frac{8 \text{tnw}}{4.5 \text{tbck}} - 1 \right)$$

#W = Number of Wait States  
 tnw = Network Clock Period  
 tbck = BSCCK Period

The number of allowable wait states in word mode can be calculated using:

$$\#W_{(\text{word mode})} = \left( \frac{5 \text{tnw}}{2 \text{tbck}} - 1 \right)$$

### 15.0 Switching Characteristics (Continued)



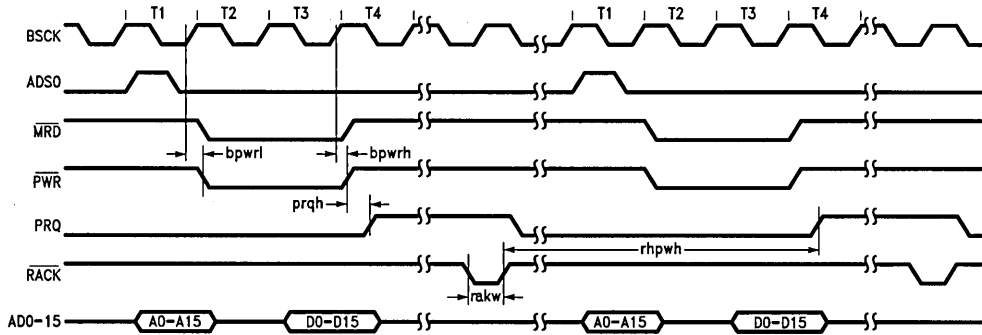
TL/F/8582-84

Symbol	Parameter	Min	Max	Units
bpwrl	Bus Clock to Port Write Low		43	ns
bpwrh	Bus Clock to Port Write High		40	ns
prqh	Port Write High to Port Request High (Note 1)		30	ns
prql	Port Request Low from Read Acknowledge High		45	ns
rakw	Remote Acknowledge Read Strobe Pulse Width	20		ns

**Note 1:** Start of next transfer is dependent on where RACK is generated relative to BSCCK and whether a local DMA is pending.

### 15.0 Switching Characteristics (Continued)

Remote DMA (Read, Send Command) Recovery Time



TL/F/8582-85

Symbol	Parameter	Min	Max	Units
bpwrl	Bus Clock to Port Write Low		43	ns
bpwrh	Bus Clock to Port Write High		40	ns
prqh	Port Write High to Port Request High (Note 1)		30	ns
prql	Port Request Low from Read Acknowledge High		45	ns
rakw	Remote Acknowledge Read Strobe Pulse Width	20		ns
rhpwh	Read Acknowledge High to Next Port Write Cycle (Notes 2,3,4)	11		BUSCK

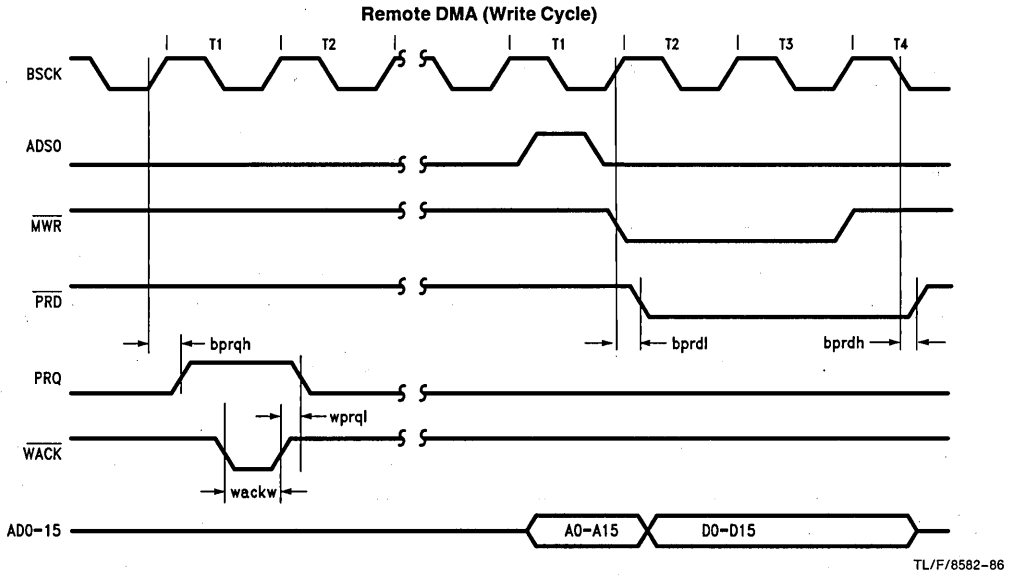
**Note 1:** Start of next transfer is dependent on where  $\overline{RACK}$  is generated relative to BSCCK and whether a local DMA is pending.

**Note 2:** This is not a measured value but guaranteed by design.

**Note 3:**  $\overline{RACK}$  must be high for a minimum of 7 BUSCK.

**Note 4:** Assumes no local DMA interleave, no  $\overline{CS}$ , and immediate BACK.

### 15.0 Switching Characteristics (Continued)



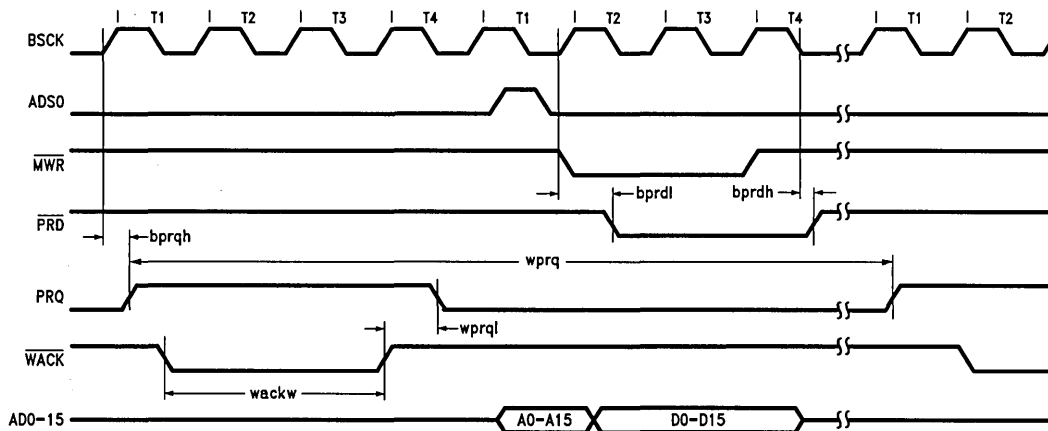
Symbol	Parameter	Min	Max	Units
bprqh	Bus Clock to Port Request High (Note 1)		42	ns
wprql	WACK to Port Request Low		45	ns
wackw	WACK Pulse Width	20		ns
bprdl	Bus Clock to Port Read Low (Note 2)		40	ns
bprdh	Bus Clock to Port Read High		40	ns

**Note 1:** The first port request is issued in response to the remote write command. It is subsequently issued on T1 clock cycles following completion of remote DMA cycles.

**Note 2:** The start of the remote DMA write following WACK is dependent on where WACK is issued relative to BUSCK and whether a local DMA is pending.

## 15.0 Switching Characteristics (Continued)

Remote DMA (Write Cycle) Recovery Time



TL/F/8582-87

Symbol	Parameter	Min	Max	Units
bprqh	Bus Clock to Port Request High (Note 1)		40	ns
wprql	$\overline{WACK}$ to Port Request Low		45	ns
wackw	$\overline{WACK}$ Pulse Width	20		ns
bprdl	Bus Clock to Port Read Low (Note 2)		40	ns
bprdh	Bus Clock to Port Read High		40	ns
wprq	Remote Write Port Request to Port Request Time (Notes 3,4,5)	12		BUSCK

**Note 1:** The first port request is issued in response to the remote write command. It is subsequently issued on T1 clock cycles following completion of remote DMA cycles.

**Note 2:** The start of the remote DMA write following  $\overline{WACK}$  is dependent on where  $\overline{WACK}$  is issued relative to BUSCK and whether a local DMA is pending.

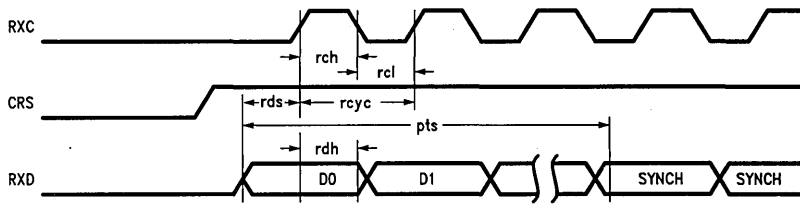
**Note 3:** Assuming wackw < 1 BUSCK, and no local DMA interleave, no  $\overline{CS}$ , immediate  $\overline{BACR}$ , and  $\overline{WACK}$  goes high before T4.

**Note 4:**  $\overline{WACK}$  must be high for a minimum of 7 BUSCK.

**Note 5:** This is not a measured value but guaranteed by design.

## 15.0 Switching Characteristics (Continued)

Serial Timing—Receive (Beginning of Frame)



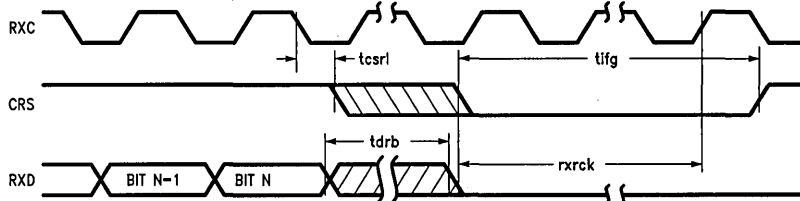
TL/F/8582-88

Symbol	Parameter	Min	Max	Units
rch	Receive Clock High Time	40		ns
rcl	Receive Clock Low Time	40		ns
rcyc	Receive Clock Cycle Time	80	120	ns
rds	Receive Data Setup Time to Receive Clock High (Note 1)	20		ns
rdh	Receive Data Hold Time from Receive Clock High	17		ns
pts	First Preamble Bit to Synch (Note 2)	8		rcyc cycles

**Note 1:** All bits entering NIC must be properly decoded, if the PLL is still locking, the clock to the NIC should be disabled or CRS delayed. Any two sequential 1 data bits will be interpreted as Synch.

**Note 2:** This is a minimum requirement which allows reception of a packet.

Serial Timing—Receive (End of Frame)



TL/F/8582-89

Symbol	Parameter	Min	Max	Units
rxrck	Minimum Number of Receive Clocks after CRS Low (Note 1)	5		rcyc cycles
tdrb	Maximum of Allowed Dribble Bits/Clocks (Note 2)		3	rcyc cycles
tifg	Receive Recovery Time (Notes 4,5)		40	rcyc cycles
tcsl	Receive Clock to Carrier Sense Low (Note 3)	0	1	rcyc cycles

**Note 1:** The NIC requires a minimum number of receive clocks following the de-assertion of carrier sense (CRS). These additional clocks are provided by the DP8391 SNI. If other decoder/PLLs are being used additional clocks should be provided. Short clocks or glitches are not allowed.

**Note 2:** Up to 5 bits of dribble bits can be tolerated without resulting in a receive error.

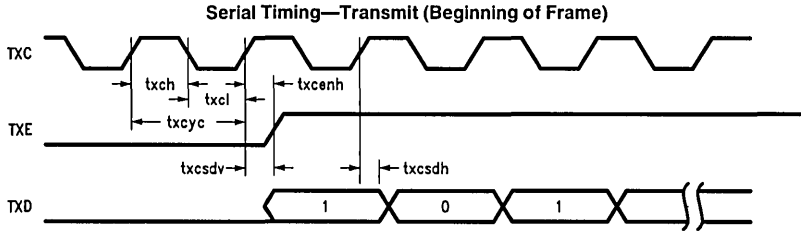
**Note 3:** Guarantees to only load bit N, additional bits up to tdrb can be tolerated.

**Note 4:** This is the time required for the receive state machine to complete end of receive processing. This parameter is not measured but is guaranteed by design. This is not a measured parameter but is a design requirement.

**Note 5:** CRS must remain de-asserted for a minimum of 2 RXC cycles to be recognized as end of carrier.



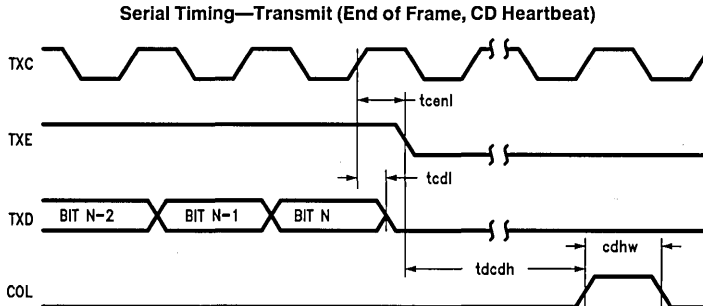
## 15.0 Switching Characteristics (Continued)



TL/F/8582-90

Symbol	Parameter	Min	Max	Units
txch	Transmit Clock High Time	36		ns
txcl	Transmit Clock Low Time	36		ns
txcyc	Transmit Clock Cycle Time	80	120	ns
txcenh	Transmit Clock to Transmit Enable High (Note 1)		48	ns
txcsdv	Transmit Clock to Serial Data Valid		67	ns
txcsdh	Serial Data Hold Time from Transmit Clock High	10		ns

**Note 1:** The NIC issues TXEN coincident with the first bit of preamble. The first bit of preamble is always a 1.

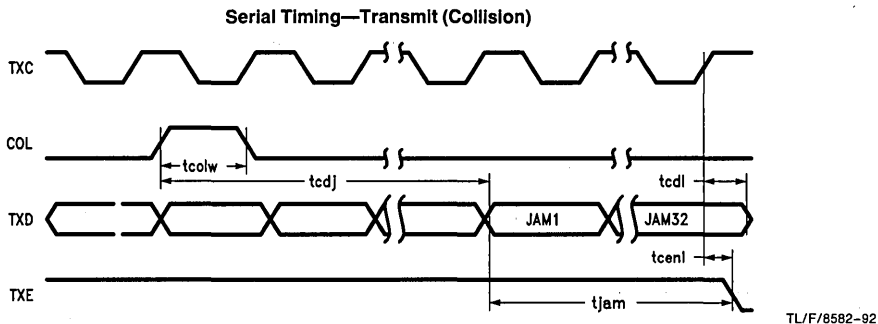


TL/F/8582-91

Symbol	Parameter	Min	Max	Units
tcdl	Transmit Clock to Data Low		55	ns
tcenl	Transmit Clock to TXEN Low		55	ns
tdcdh	TXEN Low to Start of Collision Detect Heartbeat (Note 1)	0	64	txcyc cycles
cdhw	Collision Detect Width	2		txcyc cycles

**Note 1:** If COL is not seen during the first 64 TX clock cycles following de-assertion of TXEN, the CDH bit in the TSR is set.

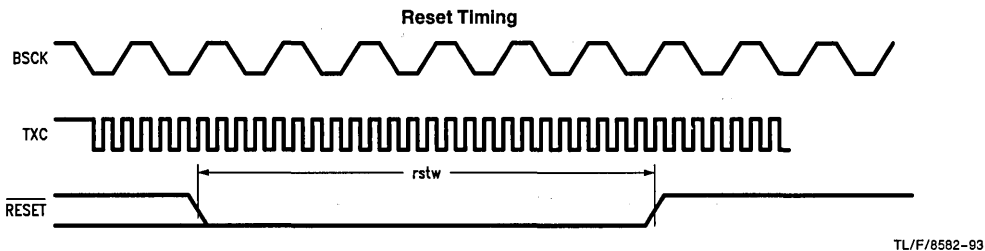
## 15.0 Switching Characteristics (Continued)



Symbol	Parameter	Min	Max	Units
tcolw	Collision Detect Width	2		txcyc. cycles
tcdj	Delay from Collision to First Bit of Jam (Note 1)		8	txcyc. cycles
tjam	Jam Period (Note 2)		32	txcyc. cycles

**Note 1:** The NIC must synchronize to collision detect. If the NIC is in the middle of serializing a byte of data the remainder of the byte will be serialized. Thus the jam pattern will start anywhere from 1 to 8 TXC cycles after COL is asserted.

**Note 2:** The NIC always issues 32 bits of jam. The jam is all 1's data.



Symbol	Parameter	Min	Max	Units
rstw	Reset Pulse Width (Note 1)	8		BSCK Cycles or TXC Cycles (Note 2)

**Note 1:** The RESET pulse requires that BSCK and TXC be stable. On power up, RESET should not be raised until BSCK and TXC have become stable. Several registers are affected by RESET. Consult the register descriptions for details.

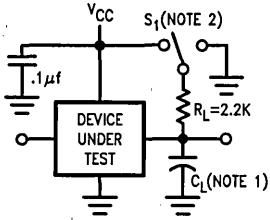
**Note 2:** The slower of BSCK or TXC clocks will determine the minimum time for the RESET signal to be low.

If BSCK < TXC then RESET = 8 × BSCK

If TXC < BSCK then RESET = 8 × TXC

## AC Timing Test Conditions

Input Pulse Levels	GND to 3.0V
Input Rise and Fall Times	5 ns
Input and Output Reference Levels	1.3V
TRI-STATE Reference Levels	Float ( $\Delta V$ ) $\pm 0.5V$
Output Load (See Figure below)	



TL/F/8582-94

**Note 1:**  $C_L = 50$  pF, includes scope and jig capacitance.

**Note 2:** S1 = Open for timing tests for push pull outputs.

S1 =  $V_{CC}$  for  $V_{OL}$  test.

S1 = GND for  $V_{OH}$  test.

S1 =  $V_{CC}$  for High Impedance to active low and active low to High Impedance measurements.

S1 = GND for High Impedance to active high and active high to High Impedance measurements.

## Pin Capacitance $T_A = 25^\circ C, f = 1$ MHz

Parameter	Description	Typ	Max	Unit
$C_{IN}$	Input Capacitance	7	15	pF
$C_{OUT}$	Output Capacitance	7	15	pF

**Note:** This parameter is sampled and not 100% tested.

### DERATING FACTOR

Output timings are measured with a purely capacitive load for 50 pF. The following correction factor can be used for other loads:

$C_L \geq 50$  pf:  $+0.3$  ns/pF (for all outputs except TXE, TXD, and LBK)

# DP8390 Network Interface Controller: An Introductory Guide

National Semiconductor  
Application Note 475



## OVERVIEW

A general description of the DP8390 Network Interface Controller (NIC) is given in this application note. The emphasis is placed on how it operates and how it can be used. This description should be read in conjunction with the DP8390 data sheet.

## 1.0 INTRODUCTION

The DP8390 Network Interface Controller provides all the Media Access Control layer functions required for transmission and reception of packets in accordance with the IEEE 802.3 CSMA/CD standard. The controller acts as an advanced peripheral and serves as a complete interface between the system and the network. The onboard FIFO and DMA channels work together to form a straight-forward packet management scheme, providing (local) DMA transfers at up to 10 megabytes per second while tolerating typical bus latencies.

A second set of DMA channels (remote DMA) is provided on chip, and is integrated into the packet management scheme to aid in the system interface. The DP8390 was designed with the popular 8, 16 and 32 bit microprocessors in mind, and gives system designers several architectural options. The NIC is fabricated using National Semiconductor's double metal 2 micron microCMOS process, yielding high speed with very low power dissipation.

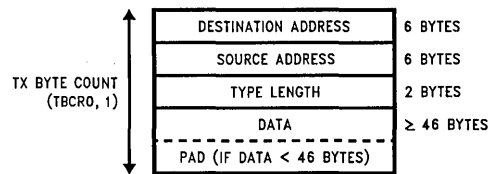
## 2.0 METHOD OF OPERATION

The NIC is used as a standard peripheral device and is controlled through an array of on-chip registers. These registers are used during initialization, packet transmission and reception, and remote DMA operations. At initialization, the physical address and multicast address filters are set, the receiver, transmitter and data paths are configured, the DMA channels are prepared, and the appropriate interrupts are masked. The Command Register (CR) is used to initiate transmission and remote DMA operations.

Upon packet reception, end of packet transmission, remote DMA completion or error conditions, an interrupt is generated to indicate that an action should be taken. The processor's interrupt driven routine then reads the Interrupt Status Register (ISR) to determine what type of interrupt occurred, and performs the appropriate actions.

## 3.0 PACKET TRANSMISSION

The NIC transmits packets in accordance with the CSMA/CD protocol, scheduling retransmission of packets up to 15 times on collisions according to the truncated binary exponential backoff algorithm. No additional processor intervention is required once the transmit command is given.

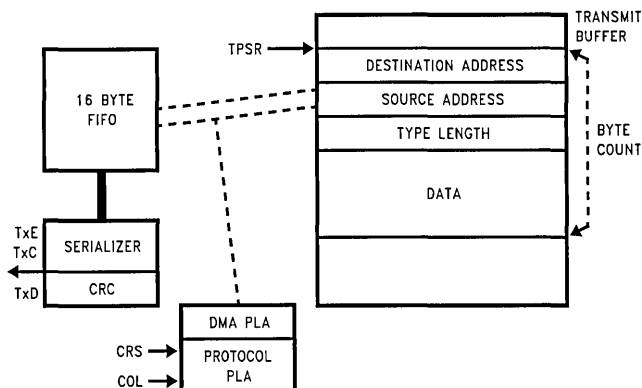


TL/F/9141-1

FIGURE 1. Transmit Packet Format

## 3.1 Transmission Setup

After a packet that conforms to the IEEE 802.3 specification is set up in memory, with 6 bytes of the destination address, followed by 6 bytes of the source address, followed by the data byte count and the data, it is ready for transmission (see Figure 1). To transmit a packet, the NIC is given the starting address of the packet (TPSR), the length of the packet (TBCRO, TBCR1), and then the PTX (transmit packet) bit of the Command Register is set to initiate the transmission (see Figure 2).



TL/F/9141-2

FIGURE 2. Packet Transmission

### 3.2 Transmission Process

Once the transmit command is given, if no reception is in progress, the transmit prefetch begins. The high speed local DMA channel bursts data into the NIC's FIFO. After the first DMA transfer of the prefetch burst, if no carrier is present on the network, and the NIC is not deferring, the TXE (transmit enable) signal is asserted and the transmission begins. After the 62 bits of preamble (alternating ONES and ZEROS) and the start of frame delimiter (two ONES) are sent out, the data in the FIFO is serialized, and sent out as NRZ data (pin TxD) with a clock (TxC), while the CRC is calculated. When the FIFO reaches a threshold (X bytes empty) a new DMA burst is initiated. This process continues until the byte count (TBCR0 and TBCR1) reaches zero. After the last byte is serialized, the four bytes of the calculated CRC are serialized and appended to complete the packet.

Should a collision occur, the current transmission stops, a jam sequence (32 Ones) transmitted (to ensure that every node senses a collision), and a retransmission of the packet is scheduled according to the truncated Binary Exponential Backoff Routine.

### 3.3 Transmission Status

After the transmission is complete, an interrupt is generated and either the PTX bit (complete packet transmitted) or the TXE bit (packet transmission aborted) of the ISR (Interrupt Status Register) is set. The interrupt driven routine then reads the RSR (Receive Status Register) and TSR (Transmit Status Register) to find out details of the transmission. If the PTX bit is set, the RSR will reveal if a carrier was present when the transmission was initiated (DFR). The TSR will identify if the carrier was lost during the transmission (CRS—this would point to a short somewhere on the network), if the collision detect circuitry is working properly (CDH), and if collision occurred (COL). Whenever a collision is encountered during transmission, the collision count register (NCR) is incremented. Should a collision occur outside the 512 bit window (slot time), the OWC (Out of Window Collision) bit of the TSR is set.

The TXE bit of the ISR is set if 16 collisions or a FIFO underrun occurs. If the transmission is aborted due to 16 collisions, the ABT bit of the TSR is set. (If this occurs it is likely that there is an open somewhere on the network.) If the local DMA channel can not fill the FIFO faster than data is sent to the network, the FU bit (FIFO Underrun) of the TSR is set and the transmission is also aborted. This is a result of a system bandwidth problem and points to a system design flaw. System bandwidth considerations are discussed further in Section 5.1.3.

## 4.0 PACKET RECEPTION

The bus topology used in CSMA/CD networks allows every node to receive every packet transmitted on the network. The receive filters determine which packets will be buffered to memory. Since every packet is not of interest, only packets having a destination address that passes the node's receive filters will be transferred into memory. The NIC offers many options for the receive filters and implements a complete packet management scheme for storage of incoming packets.

### 4.1 Reception Process

When a carrier is first sensed on the network (i.e. CRS signal is active), the controller sees the alternating ONE - ZERO preamble and begins checking for two consecutive ONES, denoting the start of frame delimiter (SFD). Once the SFD is detected, the serial stream of data is deserialized and pushed into the FIFO, a byte at a time. As the data is being transferred into the FIFO, the first six bytes are checked against the receive address filters. If an address match occurs, the packet is DMAed from the FIFO into the receive buffer ring. If the address does not match, the packet is not buffered and the FIFO is reset.

Each time the FIFO threshold is reached, a DMA burst begins and continues for the proper number of transfers. DMA bursts continue until the end of the packet (Section 5.1.2). At the end of a reception, the NIC prepares for an immediate reception while writing the status of the previous reception to memory. An interrupt is issued to indicate that a packet was received, and is ready to be processed.

The CRC generator is free running and is reset whenever the SFD is detected. At every byte boundary the calculated value of the CRC is compared with the last four received bytes. When the CRS signal goes LOW, denoting the end of a packet, if the calculated CRC matches the received CRC on the last byte boundary, the packet is a good packet and is accepted. However, if the calculated and received CRCs do not match on the last byte boundary before CRS goes LOW, a CRC error is flagged (CRC bit of RSR set) and the packet is rejected, i.e. the receive buffer ring pointer (CURR) is not updated (Section 4.5). If the CRS signal does not go LOW on a byte boundary and a CRC error occurs, the incoming packet is misaligned, and a frame alignment error is flagged (FAE bit of RSR set). Frame alignment errors only occur with CRC errors.

### 4.2 Address Matches

The first bit received after the SFD indicates whether the incoming packet has a physical or multicast address. A ZERO indicates a physical address, that is, a unique map-

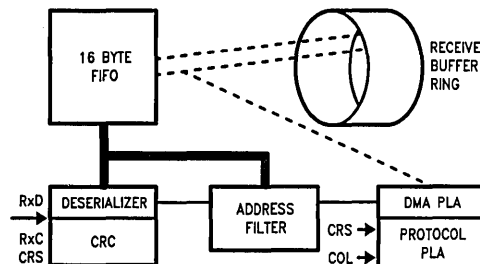


FIGURE 3. Packet Reception

TL/F/9141-3

ping between the received address and the node's 48 bit physical address as programmed at initialization (PAR0-PAR5). A ONE indicates a multicast address, meaning a packet intended for more than one node.

Multicast addressing is useful where one node needs to send a packet to multiple nodes, as in a query command. Multicast addressing provides a very fast way to perform address filtering in real time, by using an on-chip hashing table. A hashing algorithm based on the CRC is used to map the multicast address into the 64 bit Multicast Address Filter (MAF0-7).

After the CRC has been calculated on the destination address, the upper six bits of the CRC are used as an index into the Multicast Address Filter (MAF). If the selected filter bit is ONE, the packet is accepted, if the MAF bit is ZERO the packet is not accepted.

A special multicast address is the broadcast address, which denotes a packet intended to be received by all nodes. The broadcast packet has an address of all ONES (this address also maps into a bit in the MAF).

The DP8390 also provides the ability to accept all packets on the network with a physical address. Promiscuous physical mode causes any packet with a physical address to be buffered into memory. To receive all multicast packets it is necessary to set all of the MAF bits to ONE.

#### 4.3 Network Statistics

Three eight bit counters are provided for monitoring receive packet errors. After an address match occurs if a Frame Alignment or CRC error occurs, or if a packet is lost due to insufficient buffer resources (see below), the appropriate counter is incremented. These counters are cleared when read. The counters trigger an interrupt when they reach a value of 128 (if not masked) to force the processor to read (and thus clear) their contents. The counters have a maximum value of 192, providing a large latency between when the interrupt is asserted and when the counter overflows. When a CNT interrupt occurs, all three tally counters should be read and added into larger counters maintained by the processor.

#### 4.4 Setting the Receive Configuration Register

The Receive Configuration Register (RCR) is used in conjunction with the physical and multicast addresses to determine which packets should be accepted and placed in the receive buffer ring. The RCR is initialized to accept physical, multicast and/or broadcast packets, or alternatively to place the receiver in promiscuous mode to accept all packets with a physical address. If the MON bit of the RCR is set, placing the receiver in monitor mode, the receiver still checks the addresses of incoming packets according to the set up address filter, and network statistics are still gathered, but packets are not buffered into memory.

The minimum packet size in standard 802.3 networks is 64 bytes long. Packets less than 64 bytes are considered runt packets and are normally rejected. However, in some applications it may be desirable to accept such packets. By setting the AR bit of the RCR, runt packets are accepted.

For diagnostic purposes it may be desirable to examine errored packets, and not overwrite them with good packets as is done in normal operation. By setting the SEP bit of the RCR, errored packets are saved and their status is written to memory.

#### 4.5 Receive Buffer Ring

As packets are received they are placed into the receive buffer ring, and as they are processed they are removed from this ring. At initialization, an area of memory is allocated to act as the receive buffer ring, and the NIC's buffer management scheme then makes efficient use of this memory. The ring pointers are contained on chip and the DMA channels can work at up to a 10 Mbyte/sec transfer rate. A second DMA channel, the remote DMA channel, is available for transferring packets out of the receive buffer ring.

The buffer management scheme effectively works as a large packet FIFO and is very appropriate for most networking applications because packets are generally processed in the order they are received.

Four pointers are used to control the ring; the page start (PSTART) and page stop (PSTOP) pointers determine the size of the buffer ring, the current page (CURR) pointer determines where the next packet will be loaded,

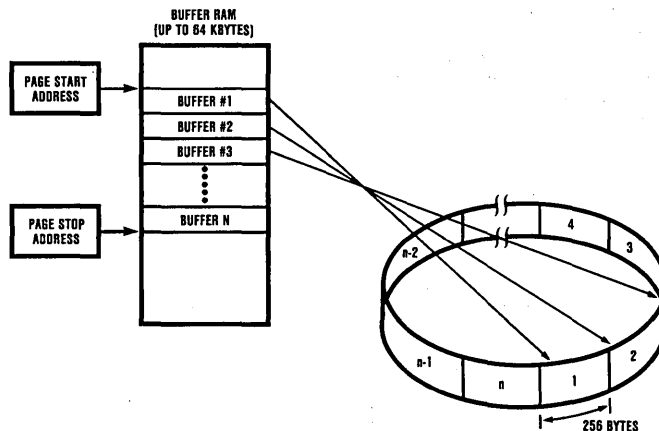
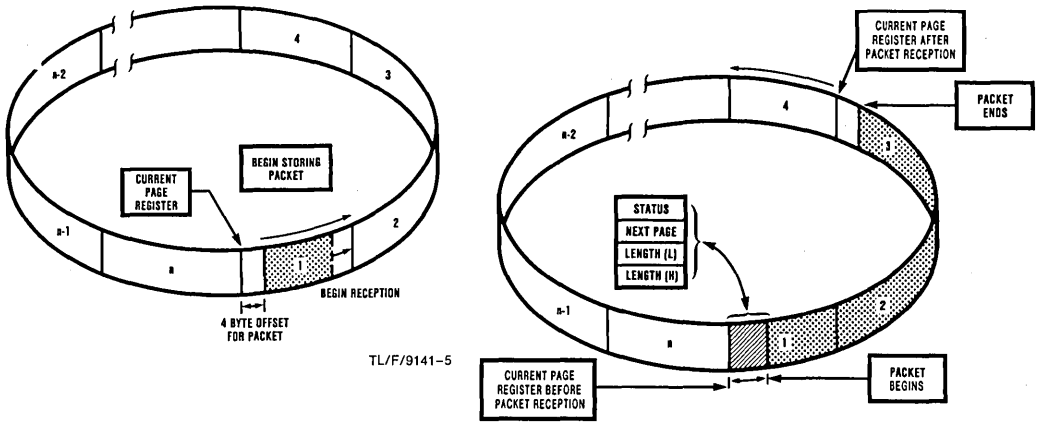


FIGURE 4. The Receive Buffer

TL/F/9141-4



TL/F/9141-5

TL/F/9141-6

FIGURE 5. Receive Packet Buffering

and the boundary (BNRY) pointer indicates where the next packet to be unloaded (or processed) begins. As packets are received, the current pointer moves ahead of the boundary pointer around the ring. The page start and stop pointers remain unchanged during operation.

The receive buffer ring is divided into 256 byte buffers, and these buffers are linked together as required by the received packets (see Figure 4). Up to 256 of these buffers can be linked together in the receive buffer ring, yielding a maximum buffer size of 64K bytes. Since all NIC registers are 8 bits wide, the ring pointers refer to 256 byte boundaries within a 64K byte space.

At initialization, PSTART register is loaded with the beginning page address of the ring, and PSTOP is loaded with the ending page address of the ring.

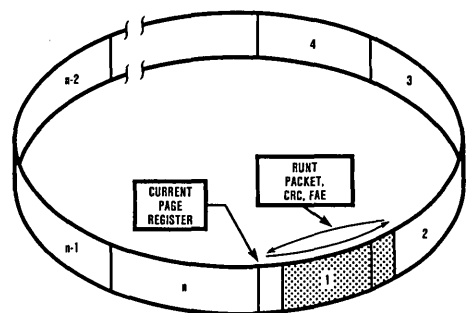
On a valid reception, the packet is placed in the ring at the page pointed to by CURR plus a 4 byte offset (see Figure 5). The packet is transferred to the ring, a DMA burst at a time. When necessary, buffers are automatically linked together, until the complete packet is received. The last and first buffers of the ring buffer are linked just as the first and second buffers. At the end of a reception, the status from the Receive Status Register (RSR), a pointer to the next

packet, and the byte count of the current packet are written into the 4 byte offset.

If a receive error occurs (FAE, CRC) CURR is not updated at the end of a reception, so the next packet received overwrites the bad packet (see Figure 6). This feature can be disabled (by setting the save errored packet (SEP) bit in the RCR) to allow examination of errored packets.

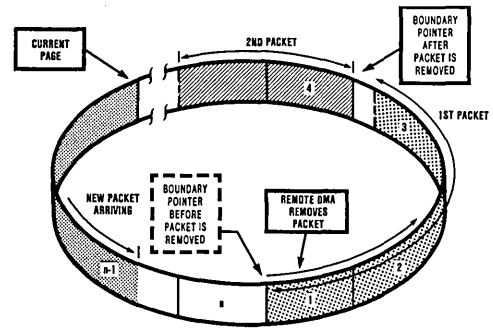
At receiving nodes, collision fragments may be seen as runt packets. A runt packet is a packet less than 64 bytes (512 bits) long, and since a collision must occur in the first 512 bit times, the packet will be truncated to less than 64 bytes. After runt packets are received, the CURR is not updated, so the next packet received will overwrite the runt packet. This standard feature can be suppressed by setting the AR bit in the TCR. This is useful when it is desirable to examine collision fragments, and in non-standard applications where smaller packets are used.

Once packets are in the receive ring they must be processed. However, the amount of processing that occurs while the packet is in the buffer ring varies according to the implementation. As packets are removed from the buffer ring, the boundary pointer (BNRY) must be updated. The BNRY always follows CURR around the ring (see Figure 7).



TL/F/9141-7

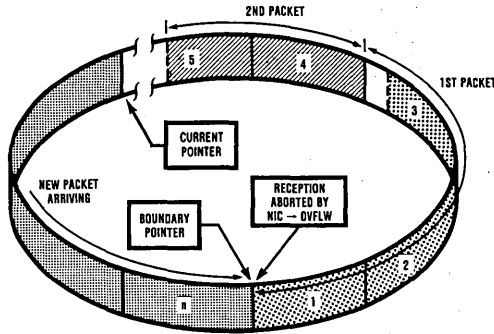
FIGURE 6. Packet Rejection



TL/F/9141-8

FIGURE 7. Removing Packets From Receive Buffer Ring

If the current local DMA address ever reaches BNR<sub>Y</sub>, the ring is full. In this case, the current and any additional receptions are aborted and tallied until the BNR<sub>Y</sub> pointer is updated. Packets already present in the ring will not be overwritten (see Figure 8). All missed packets will increment the missed packet tally counter. When enough memory is allocated for the receive buffer ring, the overwrite warning (setting of the OVW bit of the ISR) should seldom occur.



TL/F/9141-9

FIGURE 8. Receive Buffer Ring Overwrite Protection

A second set of DMA channels has been included on the DP8390 to aid in the transfer of packets out of the buffer ring. These Remote DMA channels can work in close co-operation with the receive buffer ring to provide a very effective system interface.

If the BNR<sub>Y</sub> is placed outside of the buffer ring, no overwrite protection will be present, and incoming packets may overwrite packets that have not been processed. This may be useful when evaluating the DP8390, but in normal operation it is not recommended.

5.0 SYSTEM/NETWORK INTERFACE

The DP8390 offers considerable flexibility when designing a system/network interface. This flexibility allows the designer to choose the appropriate price/performance combination while easing the actual design process.

5.1 Interfacing Considerations

Several features have been included on the NIC to allow it to easily be integrated into many systems. The size of the data paths, the byte ordering, and the bus latencies are all programmable. In addition, the clock used for the DMA channels is not coupled to the network clock, so the NIC's DMA can easily be integrated into memory systems.

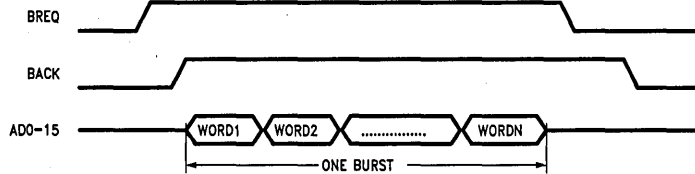
5.1.1 Data Path

The NIC can interface with 8, 16, and 32 bit microprocessors. The data paths are configurable for both byte-wide and word-wide transfers (bit WTS in DCR). When in word-wide mode, the byte ordering is programmable to accommodate both popular byte ordering schemes. All NIC registers are 8 bits wide to allow 8, 16 and 32 bit processors to access them with no additional hardware. If the NIC's 16 address lines (64K bytes) do not provide an adequate address space, the two DMA channels can be concatenated to form a 32 bit DMA address (bit LAS in DCR).

5.1.2 Local DMA

The DMA transfers between the FIFO and memory during transmission and reception occur in bursts. The bursts begin when the FIFO threshold is reached. Since only a single FIFO is required (because a node cannot receive and transmit simultaneously), the threshold takes on different meanings during transmission and reception. During reception the FIFO threshold refers to the number of bytes in the FIFO. During transmission the FIFO threshold refers to the number of empty bytes in the FIFO (16 - # bytes in FIFO). The FIFO threshold is set to 2, 4, 8 or 12 bytes (1, 2, 4 or 6 words) in the DCR (bits FT0, FT1).

The number of transfers that occur in a burst is equal to the FIFO threshold (see Figure 9).



where N = 1, 2, 4 or 6 Words or N = 2, 4, 8, or 12 Bytes when in byte mode

TL/F/9141-10

FIGURE 9. Local DMA Burst



Before a burst can begin, the NIC must first arbitrate to become master of the bus. It requests the bus by activating the BREQ signal and waiting for acknowledgment with the BACK signal. Once the NIC becomes the master of the bus, the byte/word transfers may begin. The frequency of the DMA clock is not related to the network clock, and can be input (pin 25) as any frequency up to 20 MHz. For 10 Mbit/sec networks the DMA clock can be as slow as 6 MHz. This allows tailoring of the DMA channel, to the system. The local DMA channel can burst data into and out of the FIFO at up to 10 Mbyte/sec (8X the speed of standard Ethernet). This means that during transmission or reception the network interface could require as little as one eighth of the bus bandwidth.

### 5.1.3 Bus Analysis

Two parameters useful in analysis of bus systems are the Bus Latency and the Bus Utilization. The Bus Latency is the maximum time between the NIC assertion of BREQ and the system granting of BACK. This is of importance because of the finite size of the NIC's internal FIFO. If the bus latency becomes too great, the FIFO overflows during reception (FIFO overrun error) or underflows during transmission (FIFO underrun error). Both conditions result in an error that aborts the reception or transmission. In a well designed system these errors should never occur. The Bus Utilization is the fraction of time the NIC is the master of the bus. It is desirable to minimize the time the NIC occupies the bus, in order to maximize its use by the rest of the system. When designing a system it is necessary to guarantee the NIC a certain Bus Latency, and it is desirable to minimize the Bus Utilization required by the NIC.

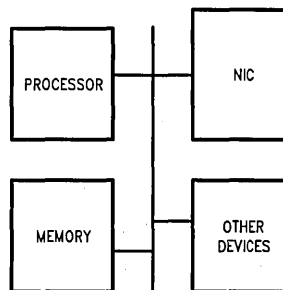
Associated with each DMA burst is a DMA set up and recovery time. When a packet is being transferred either to or from memory it will be transferred in a series of bursts. If more byte/word transfers are accomplished in each burst, fewer bursts are required to transfer the complete packet, and less time is spent on DMA set up and recovery. Thus, when longer bursts are used, less bus bandwidth is required to complete the same packet transfer.

## 6.0 INTERFACE OPTIONS

The network interface can be incorporated into systems in several ways. The network interface can be controlled by either a system processor or a dedicated processor, and can utilize either system memory or buffer memory. This section covers the basic interface architectures.

### 6.1 Single Bus System

The least complex implementation places the NIC on the same bus as the processor (see *Figure 10*). The DP8390 acts as both a master and a slave on this bus; a master during DMA bursts, and a slave during NIC register accesses. This architecture is commonly seen on motherboards in personal computers and low cost workstations, but until recently without an integrated network interface. A major issue in such designs is the bus bandwidth for use by the processor. The DP8390 is particularly suitable for such applications because of its bus utilization characteristics. During transmissions and receptions, the only time the NIC becomes a bus master, the DP8390 can require as little as one-eighth the bus bandwidth. In addition, the bus tailoring features ease its integration into such systems.



TL/F/9141-11

**FIGURE 10. Single Bus Configuration**

The design must be able to guarantee the NIC a maximum bus latency ( $< 1.6 \mu\text{s}$  for 10 Mbit/s networks), because of the finite size of the on-chip FIFO. In bus systems where the NIC is the highest priority device, this should present no problem. However, if the bus contains other devices such as Disk, DMA and Graphic controllers that require the bus for more than  $10 \mu\text{s}$  during high priority or real time activities, meeting this maximum bus latency criteria could present a problem.

Likewise, many existing single bus systems make no provision for external devices to become bus masters, and if they do, it is only under several restrictions. In such cases, an interface without the mentioned bus latency restrictions is highly desirable.

## 6.2 Dual Port Memory

One popular method of increasing the apparent bus latency of an interface, has the added effect of shielding the system bus from the high priority network bandwidth. In this application, the Dual Port Memory (DPM) allows the system bus to access the memory through one port, while the network interface accesses it through the other port. In this way, all of the high priority network bandwidth is localized on a dedicated bus, with little effect on the system bus (see *Figure 11*).

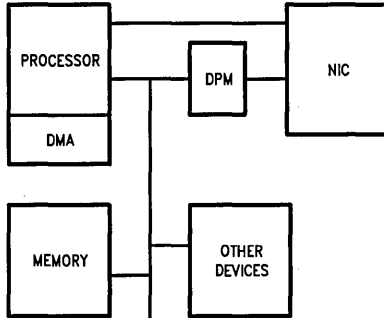


FIGURE 11. DPM Configuration

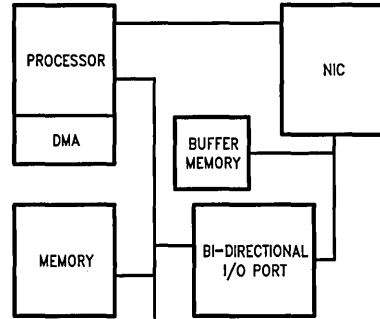
TL/F/9141-12

Dual Port Memories are typically smaller than the main memory and little, if any, processing can occur while the packets are in the DPM. Therefore, the processor (or if available, DMA controller) must transfer data between the DPM and the main memory before beginning packet processing. In this example, the DPM acts as a large packet FIFO.

Such configurations provide popular solutions. Aside from the extra complexity of the software and the DPM contention logic, higher performance can be achieved.

## 6.3 Dual Port Memory Equivalent

The functional equivalent of a Dual Port Memory implementation can be realized for low cost with the DP8390. This configuration makes use of the NIC's Remote DMA capabilities and requires only a buffer memory, and a bidirectional I/O port (see *Figure 12*). The complete network interface, with 8k x 8 of buffer memory, easily fits onto a half size IBM-PC card (as in the Network Interface Adapter, NIA, for the IBM-PC.)



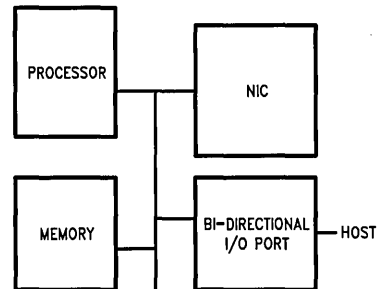
TL/F/9141-13

FIGURE 12. DPM Equivalent Configuration

The high priority network bandwidth is decoupled from the system bus, and the system interacts with the buffer memory using a lower priority bi-directional I/O port. For example, when a packet is received the local DMA channel transfers it into the buffer memory, part of which has been configured as the receive buffer ring. The remote DMA channel then transfers the packet on a byte by byte (or word by word) basis to the I/O port. At this point, as in the previous example, the processor (or if available, DMA channel), through a completely asynchronous protocol, transfers the packet into the main memory.

## 6.4 Dual Processor Configuration

For higher performance applications, it is desirable to off-load the lower-level packet processing functions from the main system (see *Figure 11*). A processor placed on a local bus with the NIC, memory and a bi-directional I/O port could accomplish these lower-level tasks, and communicate with the system processor through a higher level protocol. This processor could be responsible for sending acknowledgement packets, establishing and breaking logical links, assembling and disassembling files, executing remote procedure calls, etc.



TL/F/9141-14

FIGURE 13. Dual Processor Configuration

## 7.0 REMOTE DMA

A set of DMA channels is built into the DP8390 to aid in the system integration (as discussed above). Using a simple asynchronous protocol, the Remote DMA channels are used to transfer data between dedicated network memory, and common system memory. In normal operation, the remote DMA channels transfer data between the network memory and an I/O port, and the system transfers between the I/O port and the system memory. The system transfers are typically accomplished using either the processor, or a DMA controller.

The Remote DMA channels work in both directions: transmission packets are transferred into the network memory and received packets are transferred out of the network memory. Transfers into the network memory are known as remote write operations, and transfers out of the network memory are known as remote read operations. A special remote read operation, send packet, automatically removes a packet from the receive buffer ring.

### 7.1 Performing Remote DMA Operations

Before beginning a remote DMA operation, the controller must be informed of the network memory it will be using.

Both the starting address (RSAR0,1) and length (RBCR0,1) are set before initiating the remote DMA operation. The remote DMA operation begins by setting the appropriate bits in the Command Register (RDO-RD3). When the remote DMA operation is complete (all of the bytes transferred), the RDC bit (Remote DMA Complete) in the ISR (Interrupt Status Register) is set and the processor receives an interrupt, whereupon it takes the appropriate action. When the Send packet command is used, the controller automatically loads the starting address, and byte count (from the receive buffer ring) for the remote read operation, and upon completion updates the boundary pointer (BNRY) for the receive buffer ring. Only one remote DMA operation can be active at a time.

### 7.2 Hardware Considerations

The Remote DMA capabilities of the NIC are designed to require minimal external components and provide a simple implementation. An eight bit bi-directional port can be implemented using just two 374 latches (see the DP8390 Hardware Design Guide). All of the control circuitry is provided on the DP8390. In addition, bus arbitration with the local DMA is accomplished within the NIC in such a way as to not lock out other devices on the bus (see the DP8390 Data-sheet).

# Ethernet Network Interface Adapter for the Apple Macintosh II NuBus

National Semiconductor  
Application Note 686  
Andrew C. Pagnon



## INTRODUCTION

The gradual move in recent years towards distributed processing units with a need for these to communicate and the increasing demand for peripheral usage optimization has resulted in the fast growth of Local Area Networks. In an attempt to standardize communications between networks the International Standards Organization (ISO) has proposed a seven layer reference model called the Open System Interconnect (OSI) which provides an independent framework for all the emerging network standards. The IEEE has defined a number of these standards (802.3 to 802.6) covering the two lower layer functions, physical and data link layers.

National Semiconductor provides a three chip set which supports the Ethernet/Thin-wire Ethernet standard (a subset of IEEE802.3) the DP8390 Network Interface Controller (NIC), DP83910 Serial Network Interface (SNI), and DP8392 Coaxial Transceiver Interface (CTI).

The aim of this application note is to describe the implementation of a Macintosh II, IIx and IIcx to Ethernet/Thin-wire Ethernet interface solution using the NSC chip set. This solution takes the form of a network interface adapter card which on one side plugs into any of the six Macintosh II NuBus expansion slots and on the other supports two physical layer options, Ethernet and thin-wire Ethernet.

The board easily interfaces to the Macintosh II NuBus interface with few external components. This application note assumes the reader is familiar with NSC's Ethernet chip set and the Macintosh II NuBus.

The note begins with a hardware overview of the adapter card, and a background description of the NuBus interface. This is followed by a detailed description of hardware supported by the main sequencer/arbitrator state diagram. This covers arbitration and a detailed description of all the cycle types implemented on the card. The PAL equations and part list are included at the end of the note along with a detailed schematic and timing diagrams.

## HARDWARE OVERVIEW

The main function of this adapter card is to transfer Ethernet packet data to/from the Macintosh CPU via NuBus during LAN transmissions and receptions. The card supports a NuBus interface to the CPU and an Ethernet interface to the network. Data transfers between the interfaces are routed on the card's local bus through 8k words of shared buffer memory which temporarily stores ethernet packet data, thus decoupling data transfers across the two interfaces. The 8k buffer memory can be expanded to 32k by simply replacing the memory ICs.

Figure 1 shows a simplified block diagram of the adapter. Besides the basic DP8390 chip set this diagram illustrates the connection of the slot and cycle decode logic used to select the card, and generate read/write cycles. The arbiter controls whether the NIC or NuBus can access the buffer RAM. The RAM contains the transmit/received packet data, and the ROMs (actually one chip) contain the Ethernet Address and the Macintosh configuration information. The ad-

dress bus interface latches store the NuBus address from the multiplexed address/data bus, and the data bus interface consists of buffers and latches to assemble the 16-bit RAM buffer data into a 32-bit word for the NuBus.

## Transmission/Reception

For Ethernet transmissions the host CPU writes data into the transmit area of the adapter card buffer memory over the NuBus interface. The host CPU then sets up the NIC to transmit the data by writing to its internal registers. The NIC responds by fetching the data into its internal FIFO using its local DMA channel, from where it is sent to the SNI-CTI and onto the Ethernet cable. Once the data has been transmitted the NIC issues an interrupt back to the host CPU and sets a status bit in its internal register.

For Ethernet receptions data is loaded from the Ethernet cable into the internal FIFO of the NIC from the SNI and CTI. When a programmable threshold is reached in the FIFO, the NIC transfers the data into the receive area of the adapter card buffer memory using its local DMA channel. Once a complete packet has been loaded into memory, the NIC sets up a pointer in its internal register, issues an interrupt to the host CPU and sets a status bit in its internal register. The host processor responds by reading the packet from the adapter card memory over the NuBus interface and updating the packet pointers stored in the NICs internal register.

## General Adapter Architecture Considerations

A shared memory architecture has been chosen for this design to maximize data throughput while not adding any extra cost or intelligence on the card. The buffer memory is mapped into the NuBus address space and a NuBus slave interface plus local bus arbitration logic is implemented on the adapter card. The reasoning for this decision is given below.

The DP8390 efficiently supports an input/output port architecture, in which the adapter card makes use of the NICs Remote DMA facility to transfer network data between the buffer RAM and an input/output port interfacing to the NuBus and to the host CPU. This implementation is a slightly less expensive option than others however the throughput of the port interface is somewhat limiting, and there are no memory addressing limitations on the NuBus that would require an I/O port technique.

A bus master architecture, in which the adapter card can gain ownership of the host CPU bus and transfer data directly into system memory is significantly more costly and with the current generation of controllers will not yield significantly better performance across NuBus without going to the expense of adding an on-card processor.

Thus, using a buffer RAM that is addressed directly by the NuBus and the Ethernet Controller, provides the flexibility of reading/writing data via the NuBus at fast speeds, and since the DP8390's local DMA only utilizes a small percentage of the RAM's total access time (12%) the RAM is mostly free for NuBus activity.

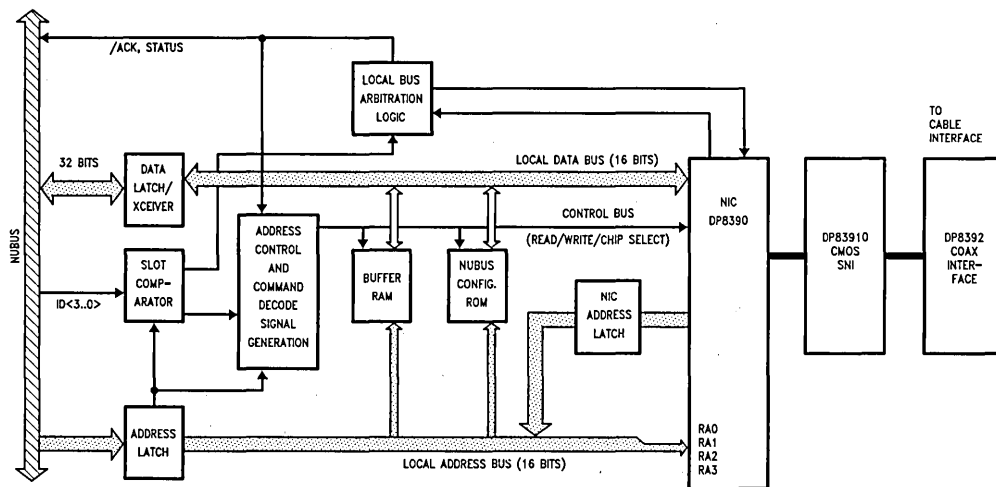


FIGURE 1. General Block Diagram for the NuBus Shared RAM Adapter

TL/F/10805-1

Once the Shared Adapter RAM approach is chosen another architectural consideration is the width of the Buffer RAM and of the CPU bus transfers supported. Table I shows the options considered. The 16-bit Buffer memory and 32-bit NuBus transfer option was chosen as the best compromise on data throughput and component count/cost.

#### General Hardware Overview

Figure 2 shows a more detailed block diagram of the Adapter. This shows each block, and details the chips used to implement each block.

The Ethernet to buffer memory interface is implemented using National Semiconductor chip set. The DP8390 Network Interface Controller is a CMOS VLSI device designed to ease interfacing with IEEE 802.3 Ethernet type Local Area Networks. It implements all Media Access Control MAC functions (a subset of the ISO data link layer) for transmission and reception of packets in accordance with the IEEE 802.3 standard. Its dual DMA channels and internal FIFO provide a simple yet efficient packet management design. All bus arbitration and memory support logic required by its dual DMA channels are integrated into the NIC.

The DP83910 Serial Network Interface is a CMOS combined analog and digital device which provides the Manchester encoding and decoding functions of IEEE 802.3 Ethernet type Local Area Networks. It contains ECL like balanced drivers and receivers, collision signal translator and a diagnostic loopback circuit.

The DP8392 Coaxial Transceiver Interface is a bipolar device used as a coaxial cable line driver/receiver for IEEE 802.3 Ethernet Local Area Networks. In Ethernet applications the transceiver is usually mounted within a dedicated enclosure (Media Access Unit) and connected to the SNI via a drop cable, while for Thin-wire Ethernet (low cost ver-

sion of Ethernet) the CTI is mounted on the same board as the SNI. Signal and power isolation requirements are met by placing a set of pulse transformers between the SNI and the CTI, and using a DC to DC converter to provide the CTI's -9V supply.

The adapter card supports a 32-bit NuBus interface to the host CPU, implemented using synchronous sequencer logic in the registered PAL 16R4. This interface supports transfers to the Network Interface Controller registers, the "Ethernet address/Mac configuration" ROM and the buffer memory. The two card interfaces must request use of the local bus before they can initiate a transfer to any of the on card devices. These requests are processed by arbitration logic which gives priority to the Ethernet interface.

The 256 x 8 ROM (LS471) contains the unique Physical Address assigned to each Ethernet board and the Configuration data required on each NuBus board which supplies identifying information about the board. This ROM can also contain device driver data.

The address/data interface to NuBus consists of four F651s data transceivers to transfer 32-bit data from/onto the NuBus, and three F533s and an F373 used to latch the address and the transfer mode signals onto the adapter card. Data on NuBus is inverted and byte swapped, so inverting transceivers and latches are used. An exception to this is the F373 which is a non-inverting version of the F533 used to latch AD24-31 which are then compared with the ID lines of the particular NuBus slot. Also the card performs a hardware byte swap on the NuBus data.

On NuBus transfers, an F521 8-bit comparator aids in the address decode function by determining whether or not the transfer is intended for the adapter card.

1-344

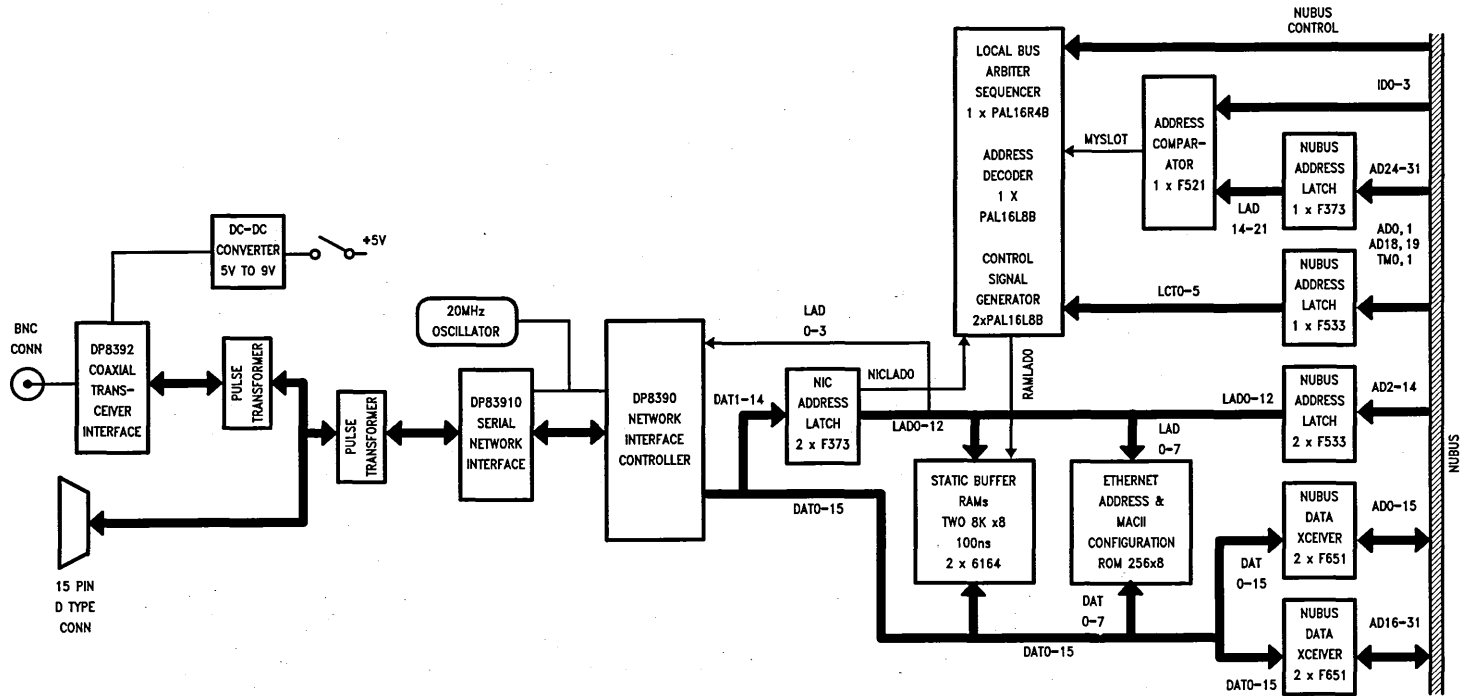


FIGURE 2. Macintosh II Ethernet Adapter Card

TL/F/10805-2

TABLE I. Adapter Card NuBus RAM Width Options

NuBus Transfer Width	Buffer Memory Width	NuBus Clock Beats	NuBus Maximum Data Rate	Percent Bus Usage	Extra Devices Required
16 Bits	16 Bits	4	32 Mbits/sec	31%	None
32 Bits	16 Bits	5	53 Mbits/sec	19%	2-Transceivers 1-PAL16L8
32 Bits	32 Bits	4	64 Mbits/sec	15.5%	2-Transceivers 2-RAM 8k x 8
32 Bits	32 Bits	3	80 Mbits/sec	12.5%	2-Transceivers 2-RAM 8k x 8 (80 ns)

TABLE II. Adapter Card Cycle Type Decoding

AD0	AD1	TM0	TM1	AD18	AD19	Cycle Type
X	X	X	L	H	L	RAM Read
X	L	L	H	H	L	RAM Write Byte 0, 1
L	L	H	H	H	L	RAM Write (Byte 0)
H	L	H	H	H	L	RAM Write (Byte 1)
X	H	X	H	H	L	No Action
X	X	X	L	H	H	ROM Read
X	X	X	H	H	H	Bus Error (ROM Write)
X	X	X	X	L	L	Bus Error (No Device)
X	X	H	X	L	H	NIC Write
X	X	L	X	L	H	NIC Read

## NUBUS BACKGROUND

This section describes the NuBus implemented in the Macintosh II expansion slots, and the Ethernet adapter card implementation of its interface. It covers NuBus' main features and signals as used by the card, followed by a description of the address space and addressing modes and ending with a description of the NuBus interface protocol.

NuBus is the bus chosen by Apple to drive the expansion slots of the Macintosh II. Its main features are:

- 32-bit wide multiplexed address data lines
- Synchronous 10 MHz clock cycle (75% duty cycle)
- Read and Write cycles (Mac II does not support block transfers)
- I/O and interrupts are memory mapped
- Geographical addressing lines

Each slot has its own geographical addressing lines onto the adapter board. This is illustrated in *Figure 3* by the super slot space in which each card has its own 256 Mbytes of memory space. Therefore no board configuration is required. (Described later.)

The Ethernet adapter card only implements a NuBus slave interface and therefore arbitration logic to gain bus mastery has not been implemented. Note also that no parity generating/checking logic has been implemented either.

The Ethernet adapter board uses the following NuBus signals:

- Clock
- Reset

- Card Slot Identification
- Non-Master Request
- Address/Data Signals/AD<31..0>
- Control Signals /TM<1..0>, /START, /ACK(knowledge)

The evaluation board supports single word transfers to the NIC registers, Address/Configuration ROM and the Buffer RAM. During the Start clock of a NuBus cycle the NuBus address and transfer mode lines are decoded on the card as shown in *Figure 3*, and as follows:

- TM<1..0> determine the type of transfer (read/write),
- AD<24..31> determine which NuBus slot is accessed.
- AD<19..18> determine which adapter card device is accessed (NIC, ROM or RAM)
- AD<15..2> are used to access a particular location within the device.
- AD<0..1> & <20..23> are ignored by the address decode

The adapter card responds to all NuBus cycles which address the card by generating an acknowledge signal ACK for one clock period and driving a status code on the transfer mode lines. Only two of the four NuBus defined transfer modes are supported by the card, transfer complete or bus error (see Table III).

TABLE III. NuBus Status Codes

TM0	TM1	Acknowledge
H	H	Transfer Complete
L	H	Bus Error

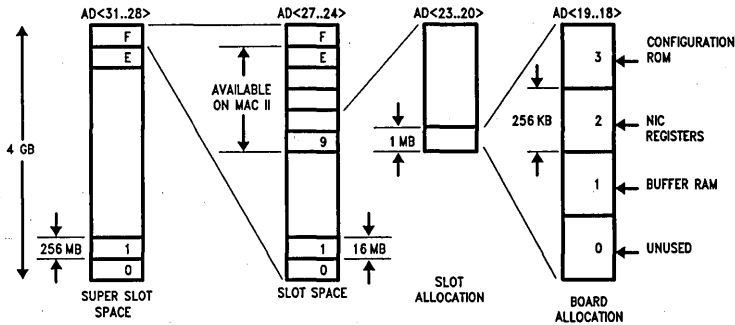


FIGURE 3. NuBus and Adapter Card Address Space Mapping

TL/F/10805-3

If the CPU requests a transfer to the adapter card NuBus slot which does not address a device on the card, or requests a write transfer to ROM, the board will respond with an error status encoding of the TM<1..0> lines during the ACK clock of the NuBus cycle. Otherwise a "Bus Transfer Complete" code is returned.

Typical NuBus read and write cycle timings are shown in Figures 4 and 5. The first NuBus cycle asserts the START line going low and presenting the address and the transfer mode. A number of clock cycles may follow before the last cycle presents the data and status on the bus, and asserting the ACK signal.

The adapter card does not implement the two other status codes supported by NuBus, "bus time out error", and "try again later". The design of the adapter card ensures that all NuBus cycles will be acknowledged within the NuBus time-out period.

**NuBus Address Space**

With a 32-bit architecture, the NuBus provides a 4 Gigabytes of address space, Figure 3. The 4 Gigabyte space is divided into sixteen 256 Megabyte Super Slots. The Super Slot being accessed is determined by decoding AD<31..28>. The top Super Slot is divided into sixteen Slot spaces by 16 Megabytes each determined by decoding AD<27..24>. Six of these slots (\$9 to \$E) are implemented as NuBus expansion board connectors on the MAC II. The interface adapter board may be plugged into any of these connectors. No hardware configuration on the adapter card is required.

**24/32 Bit Addressing Modes**

The adapter card, by ignoring address bits AD<23..20>, supports both 32- and 24-bit addressing modes.

When addressing the card in 24-bit mode, addresses of the form "\$sx xxxx" where s is the slot number can be used. The Mac II hardware translates this into a 32-bit address of the form "\$Fs0x xxxx".

When addressing the card in 32-bit mode addresses of the form "\$Fsxx xxxx" can be used. Note that as the adapter card ignores address bits AD<23..20>, addresses of the form "\$Fsxx xxxx" access the same adapter card location in both 32- and 24-bit modes, and as Apple have indicated that to ensure compatibility with future versions of the Macintosh designers should not rely on 24-bit mode addressing, it is suggested that addresses of the form "\$Fsxx xxxx" are always used.

Although supporting 24-bit mode addressing limits the memory range of each slot from 16M to 1M, this sufficiently covers the need of an Ethernet adapter card and simplifies software development. The Macintosh slot manager puts the system in 24-bit addressing mode by default and the memory manager plus some toolbox routines do not currently function properly in 32-bit mode.

**Adapter Card Address Space**

Once the slot is selected, the Network Interface Adapter's memory space is subdivided into four 256 kbyte blocks of memory determined by decoding AD<19..18>.

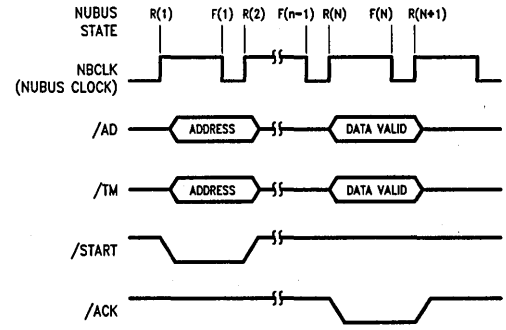


FIGURE 4. NuBus Read Cycle

TL/F/10805-4

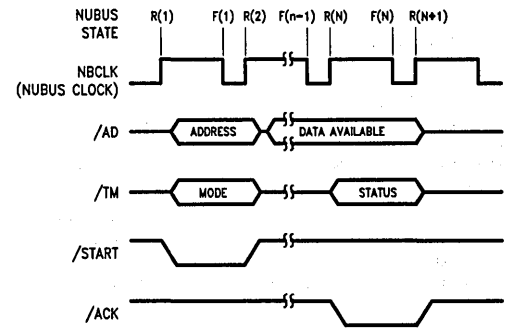


FIGURE 5. NuBus Write Cycle

TL/F/10805-5



Figure 3 shows the sub-division of the Super Slot, Slot, and Adapter Board address space.

It is important to note the following points when developing high level software to address the adapter card. If the NIC registers or the ROM are accessed as 8-bit devices, (by declaring a pointer to a character in Macintosh Programmers Workshop (MPW) for example) or as 16-bit devices (by declaring a pointer to a short integer), incrementing these pointers will usually only modify the transfer mode (by incrementing AD<1..0>) rather than increment the address to the device (NIC or RAM). It is therefore recommended that if this form of addressing is used, the RAM, ROM and NIC are declared as 32-bit devices (by accessing them with a pointer to an integer). This will ensure only word transfers take place and each NuBus address increment will also increment the address to the onboard device.

**NuBus Timing Diagrams**

The NuBus clock has a 100 ns period (10 MHz) with a 75% duty cycle (75 ns "high" and 25 ns "low"). NuBus signals are driven at the rising edge and sampled at the falling edge of the clock. A transfer (read/write cycle) is initiated when the master asserts /START, drives the address on AD<31..0>, and drives the transfer mode signals TM<1..0> with the appropriate code to indicate the desired transfer. A transfer is completed when the slave responds by asserting /ACK and driving the transfer mode signals with the appropriate status code. Please refer to Figure 4 and Figure 5 for the NuBus read/write cycles.

For a read operation, once the master has acquired the bus, a read bus transaction involves the following steps:

- R(1): The bus master asserts /START and the appropriate /ADx and /TMx lines to initiate the transfer.
- F(1): The bus slaves sample the /ADx and /TMx lines.

- R(2): The bus master releases the /ADx, /TMx, and /START lines and waits for /ACK.
- R(N): The bus slave places the requested data onto the /ADx lines, asserts /ACK, and places the appropriate status code on /TM0 and TM1 lines. (Note N may be from 2 to 256)
- F(N): The bus master samples the /ADx, /ACK, and TMx lines to receive the data and note and error condition.
- R(N + 1): The bus slave releases the /ADx and /ACK lines and the /TMx lines. This may be the R(1) transition of the next transaction.

For a write operation, once the master has acquired the bus, a write bus transaction involves the following steps:

- R(1): The bus master asserts /START and the appropriate /ADx and /TMx lines to initiate the transfer.
- F(1): The bus slaves sample the /ADx and /TMx lines.
- R(2): The bus master places the data on the /ADx lines, releases /TMx, and /START lines and waits for /ACK.
- F(2)-F(N): The bus slave samples the /ADx lines to capture the data. The data may be sampled before or during the assertion of /ACK.
- R(N): The bus slave asserts /ACK, and places the appropriate status code on /TM0 and TM1 lines when the data is accepted. (Note N may be from 2 to 256)
- F(N): The bus master samples the /ACK and TMx lines to determine the end of a transaction.
- R(N + 1): The bus master releases the /ADx while the bus slave releases the /ACK lines and the /TMx lines.

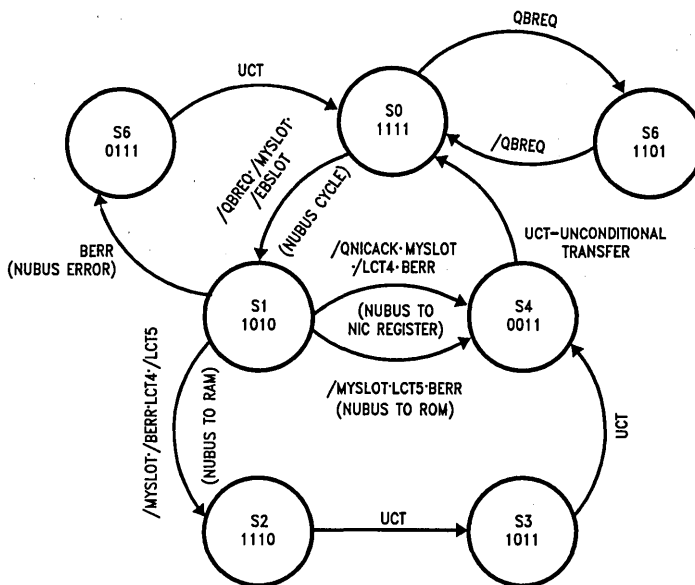


FIGURE 6. State Diagram Sequencer

TL/F/10805-6

TABLE IV. Adapter Card Cycle State Sequence for Various Cycle Types and Corresponding Diagrams

Cycle Type	State Sequence	Functional Diagram	Timing Diagram
NuBus to RAM Read	S0 → S1 → S2 → S3 → S4 → S0	Figure 12	Figure A-4
NuBus to RAM Write	S0 → S1 → S2 → S3 → S4 → S0	Figure 12	Figure A-5
NuBus to ROM Read	S0 → S1 → S4 → S0	Figure 8	Figure A-1
NuBus to NIC Read	S0 → S1 → S4 → S0	Figure 10	Figure A-2
NuBus to NIC Write	S0 → S1 → S4 → S0	Figure 10	Figure A-3
NuBus Bus Error	S0 → S1 → S5 → S0	Figure 8	
NIC to RAM Read/Write	S0 → S6 → S0	Figure 14	

#### DETAILED HARDWARE DESCRIPTION

The card's main function is to transfer Ethernet packet data from the NuBus interface to the Ethernet cable during packet transmission, and from the Ethernet cable to the NuBus interface during packet reception, via an 8k x 16 Buffer RAM, expandable to 32k x 16. In addition to this the NuBus interface is allowed direct read and write access to the NICs registers to control and monitor the NIC's operations, and read access to the "Ethernet address/Mac configuration" ROM.

This transfer of packet data from Ethernet to host CPU is executed in two distinct stages, transfers between host CPU and buffer Memory, and transfers between buffer memory and Ethernet. The former is performed by the on-card NuBus slave interface whereas the latter is performed by the NIC chip set.

A synchronous sequencer/arbitrator implemented in a PAL16R4 running on the 10 MHz NuBus clock controls all transfers supported by the adapter card. The state diagram for its operation is shown in *Figure 6*. States S1 to S5 support the NuBus slave interface, and state S6 supports the NICs interface. State S0 is the idle state. The sequence of states for each bus cycle type is shown in Table IV.

#### Arbitration for Local Card Bus by NIC/NuBus

All addressable devices on the card (Buffer RAM, NIC registers and ROM) share the common non-multiplexed local address and data buses. The two potential masters of this bus, the NuBus interface and the NIC, request access to the bus. Arbitration logic and the state sequencer resolve these requests. The sequencer only responds to master's requests during the idle state (S0). Therefore cycles already in progress are always allowed to complete before the bus is re-

allocated. Cycles always complete by returning to the idle state (S0). This prevents bus contention on the common local address bus at switchover time (NuBus/NIC) and enables arbitration to take place after every NuBus cycle or NIC local DMA burst. The NIC is given priority over the NuBus interface, so that if a NIC and a NuBus interface request are active when the sequencer is idle, the NIC cycle will be serviced first. The following bus latency discussion shows there is no real need to give one master priority over the other.

#### Bus Latency Requirements

This is defined as the time between a master issuing a request and receiving an acknowledge.

The NuBus interface bus latency allowable is determined by its 25.6  $\mu$ s bus timeout period. The maximum NuBus bus latency that can be expected on this card, that is, the longest Bus Request from the NIC, occurs if a packet ends just as the NIC performs its last FIFO burst. The local DMA burst plus the End of Packet processing operations add up to just over 4  $\mu$ s, well within the allowable 25.6  $\mu$ s.

The NIC bus latency allowable is determined by the need to prevent its internal FIFO from overflowing during packet reception. The worst case bus latency the NIC can accommodate running on a 20 MHz clock is a little more than 1  $\mu$ s (refer to the DP8390 Datasheet addendum). The maximum NIC bus latency that can be expected on this card, that is, the longest NuBus cycle to the card, is 0.5  $\mu$ s (five NuBus Clocks), well within the allowable 1  $\mu$ s.

Again, Table IV shows all the cycles supported by the card, and the state sequence followed by each one. The figures quoted display the timing diagram for each case.

## NUBUS MASTER CYCLES

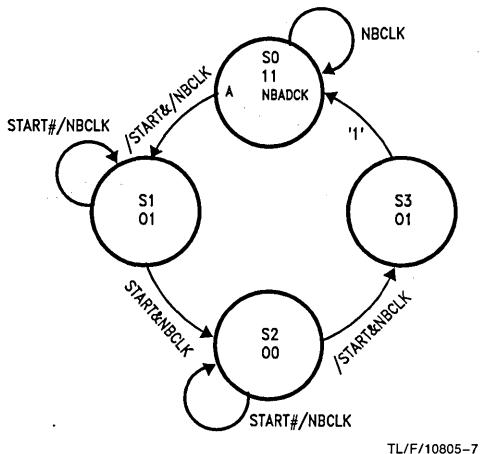
The CPU initiates NuBus cycles by driving the address and transfer mode onto NuBus and asserting the Start signal during the first clock of the bus access.

The adapter card latches the address onto the local bus on the falling edge of the first clock. A transparent latch is used so that the address is available on the local bus as soon as it is driven onto NuBus and address recognition can begin. A small asynchronous state machine in a control PAL<sup>®</sup> generates the address latching signal, which opens the latch when START becomes active and latches the address on the next falling edge of the clock. *Figure 7* shows the state machine diagram.

The address is enabled onto the local bus by NBADOE provided that the NIC is not already the bus master nor has issued a request while the sequencer is in the idle state.

The top 8 bits of the address are compared with the slot ID driven from NuBus.

The card, in response to the start signal, generates an enable signal (EBSLOT) which allows the sequencer to proceed onto the next NuBus cycle state (S1), provided the NIC bus request is not active and the above address comparison is successful (MYSLOT is active). This enable signal is cleared at the end of the NuBus cycle when ACK becomes active and prevents addresses not generated with a START signal from triggering the sequencer onto the S1 state. *Figure A-4* shows a detailed timing diagram of the address recognition operation. Note that FAST devices have been selected for the address latches and comparator to enable address recognition to meet the set up time of the sequencer.



A = State Variable (Not Used)

NBADCK = NuBus Address Clock

**FIGURE 7. State Diagram for Clock to Latch NuBus Address onto Ethernet Card**

States S1 to S4 cover the data portion of a NuBus transfer. The signal DASB (Data Strobe) is generated to qualify all the data enabling signals (NICCS, ROMCS, RAMOE, NBDBOE, DBNBOE). Note that during S2 DASB has to be released so that the state has a separate state number. This only affects NuBus to RAM cycles. Therefore during S2 the signal TOP is used to qualify the data enabling signals (RAMOE, NBDBOE, DBNBOE).

During S1 the address is further decoded with four possible outcomes. Each of the possible transfers initiated by NuBus are described.

### Bus Error Transfer

If the address is not recognized by any of the on-card devices or it is recognized by the ROM with transfer mode defining a write cycle, a bus error condition is flagged to the CPU. The sequencer enters state S5 where the DASB signal is cleared, ACK is generated to signal the end of the cycle, and a bus error code is driven onto the NuBus transfer mode lines. The sequencer then returns to idle on the next clock beat. *Figure 8* shows a timing diagram for a NuBus error cycle.

### NuBus ROM Transfer

If the address and transfer mode are decoded as a read cycle to ROM, the address decoding PAL generates a ROM enabling signal to the ROM chip select input which drives its data onto the local bus. The control PALs generate the "Data bus to NuBus output enable" signal DBNBOE to enable this data from the local bus onto NuBus. The sequencer transfers to state S4 on the next clock beat where ACK is driven onto NuBus together with the "transfer complete" code on the transfer mode signals. *Figures 8* and *A-1* show a basic and detailed timing diagram of the ROM read operation. Note that ROM set up times are easily met. Very slow ROMs can be used on this design, up to 135 ns data enable time or 210 ns address access time.

### NuBus NIC Transfer

If the address and transfer mode are decoded as a read or write cycle to the NIC register, the address decoding PAL generates the NIC chip select signal NICCS, and the bottom four bits of the local address are sent to the NIC to select one of sixteen possible NIC registers. The sequencer, *Figure 6*, remains in state S1 until the NIC generates acknowledge signal NICACK. This signal is synchronized to the NuBus clock before it is fed into the synchronous state sequencer PAL. The sequencer then proceeds onto state S4. See *Figure 9* for a functional timing diagram of NuBus to NIC read and write cycles.

If the cycle is a write, the control PAL generates the NBDBOET signal to enable the NuBus write data onto the local bus, and a small asynchronous state machine in the PAL generates the write enable signal to the NIC, SWR. *Figure 10* shows the state machine diagram. This signal is cleared on the falling edge of the clock during the S4 state to provide the necessary write data hold time to the NIC. See *Figure A-3* detailed timing diagram.

If the cycle is a read, the control PAL generates DBNBOE to enable the NIC read data from the local bus onto NuBus. See *Figure A-2* for a detailed timing diagram.

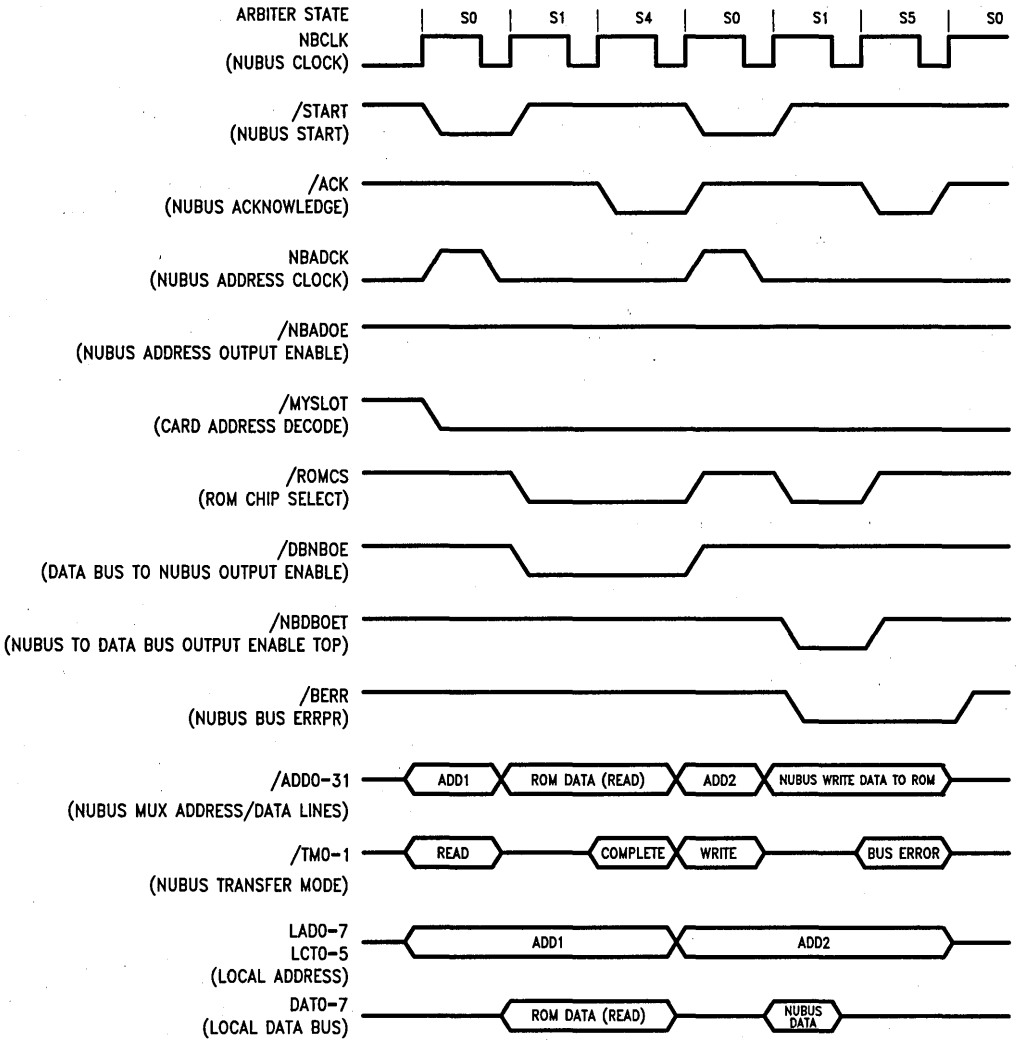


FIGURE 8. NuBus to ROM Read and Bus Error Cycles

TL/F/10805-8

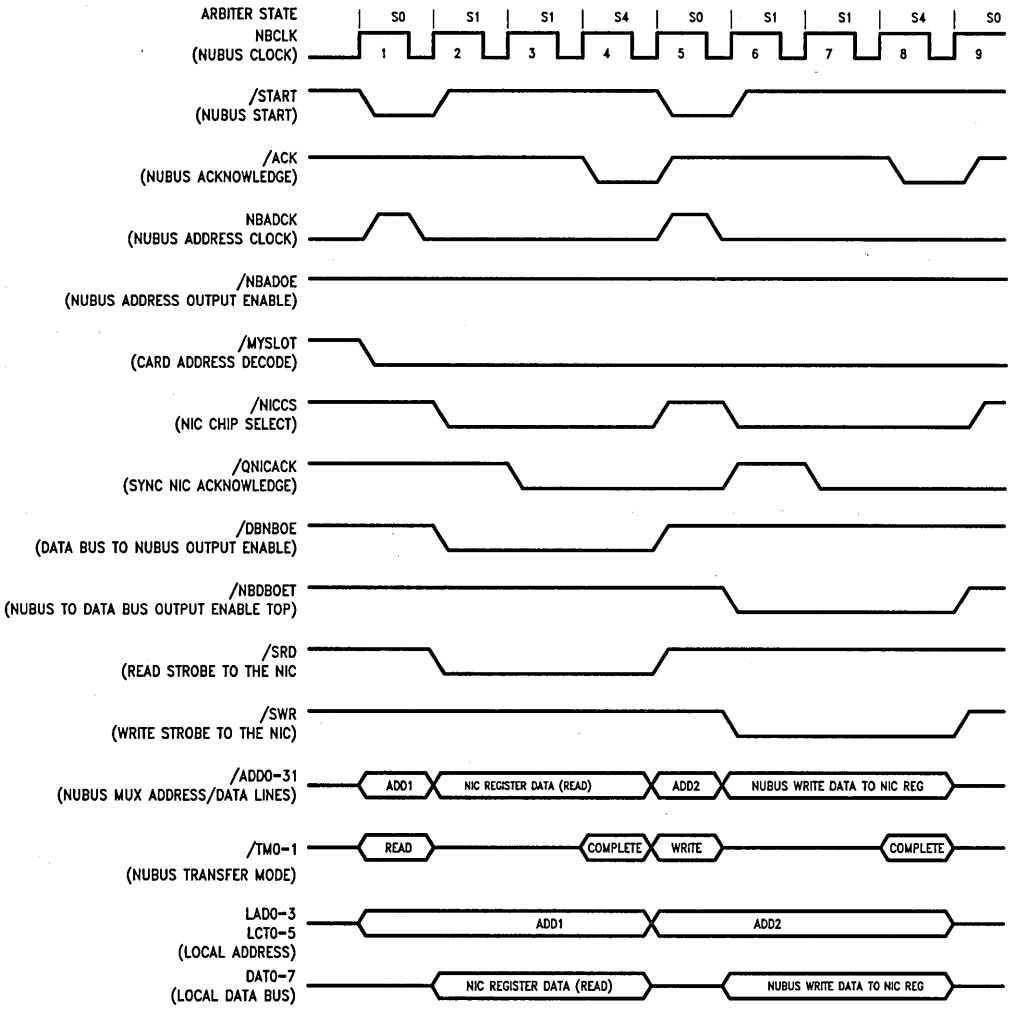
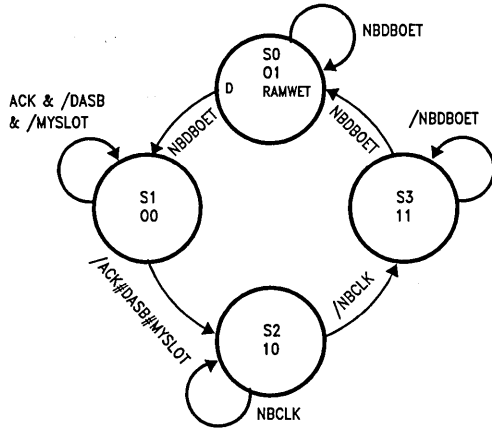


FIGURE 9. NuBus to NIC Register Read and Write Cycle

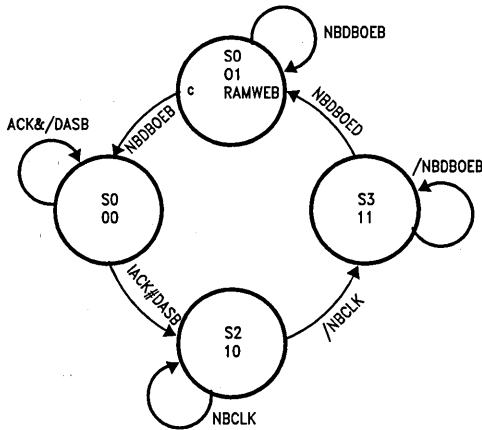
TL/F/10805-10



TL/F/10805-9

D = State Variable  
RAMWET = RAM Write Enable Top

FIGURE 10. Write Enable Top State Diagram



TL/F/10805-11

C = State Variable (Not Used)  
RAMWEB = RAM Write Enable Bottom

FIGURE 11. Write Enable Bottom State Diagram

**NuBus to RAM Transfer**

The address and transfer mode are decoded as a read or write cycle to the buffer RAM. The adapter card supports 32-bit NuBus transfers to the 16-bit buffer RAM. This is done by reading/writing to the RAM twice on every NuBus to RAM access, once during states S1 and S2 to access the least significant 16 bits of the NuBus word, and again during states S3 and S4 to access the most significant 16 bits of the NuBus word, after having incremented the bottom local address bit to the RAM (see the state diagram Figure 6). Therefore the NIC sees the buffer memory as an 8k x 16 RAM whereas the NuBus sees it as a 4k x 32 RAM. Figure 12 shows a basic timing diagram for a NuBus to RAM read and write cycle.

For read cycles' data from the RAM read during states S1 and S2 is stored in the top two NuBus transceivers by setting the transceivers in storage mode and clocking them on the falling edge of the NuBus clock during state S2 with the TBCK (Top Bus Clock) signal. This data is enabled on NuBus bits 16-31 with the signal NBDBOET. During states S3 and S4 the next RAM location is read and its data driven onto NuBus bits 0-15 through the bottom two NuBus transceivers, which are not set in storage mode (real time data mode).

Therefore by the time the adapter card drives ACK back to NuBus during state S4, the least significant 16 bits of data, corresponding to the first RAM location read, which were stored during S2, are being enabled onto NuBus bits 16-31 through the top two transceivers, and the most significant 16 bits of data, corresponding to the second RAM location read, are being enabled onto NuBus bits 0-15 through the bottom two NuBus transceivers. Figure A-4 at the end of this note, shows a detailed timing diagram of a NuBus to RAM read cycle.

Note that the adapter card performs a hardware byte swap of NuBus data through the transceivers, so that the least significant byte of data from the RAM (Bits 0-7 on the local data bus of the first RAM location read) are driven onto byte 3 of NuBus (bits 24-31). This byte will be carried on byte lane 3 in the MACII system onto byte 3 of the MC68020 (data line bits 0-7).

For write cycles, during states S1 and S2, the top two NuBus transceivers (NuBus bits 16-31) are enabled onto the local data bus and their NuBus write data written into the Buffer RAM, with the bottom local address bit clear. During the next two states S3 and S4 the bottom two NuBus transceivers (NuBus bits 0-15) are enabled onto the local data bus and their NuBus write data written into the next Buffer RAM location with the bottom local address bit set. Two separate write enable signals are generated (RAMWET and RAMWEB) and ANDed together on the card to generate RAMWE. Two small asynchronous state machines are used to generate these signals. Figure 10 and 11 show their state diagram. Figure A-5 shows a detailed timing diagram of a NuBus write cycle to RAM.

Supporting 32-bit transfers on NuBus rather than 16 only introduces one extra wait state per NuBus cycle to the RAM while doubling the data throughput per transfer.

**NIC MASTER CYCLE**

The NIC initiates local DMA cycles by driving its Bus Request line active. The sequencer/arbitrator PAL, if in idle state S0, will enter state S6 where it acknowledges the request and hands over control of the local bus to the NIC. Any requests from the NuBus interface will be held until the NIC completes its local DMA burst and clears its request line allowing the sequencer PAL to return to the idle state S0.

Figure 14 shows an NIC to RAM cycle, its request coinciding with the start of a NuBus cycle, thus illustrating the arbitration process.

Note the NIC runs on a separate 20 MHz clock, asynchronous to the NuBus clock. Therefore NIC signals to the arbitrator sequencer are first synchronized to the NuBus clock with a D-type latch (F175) before they are used by the synchronous sequencer PAL running on the NuBus clock. The signals affected are BREQ and NICACK.

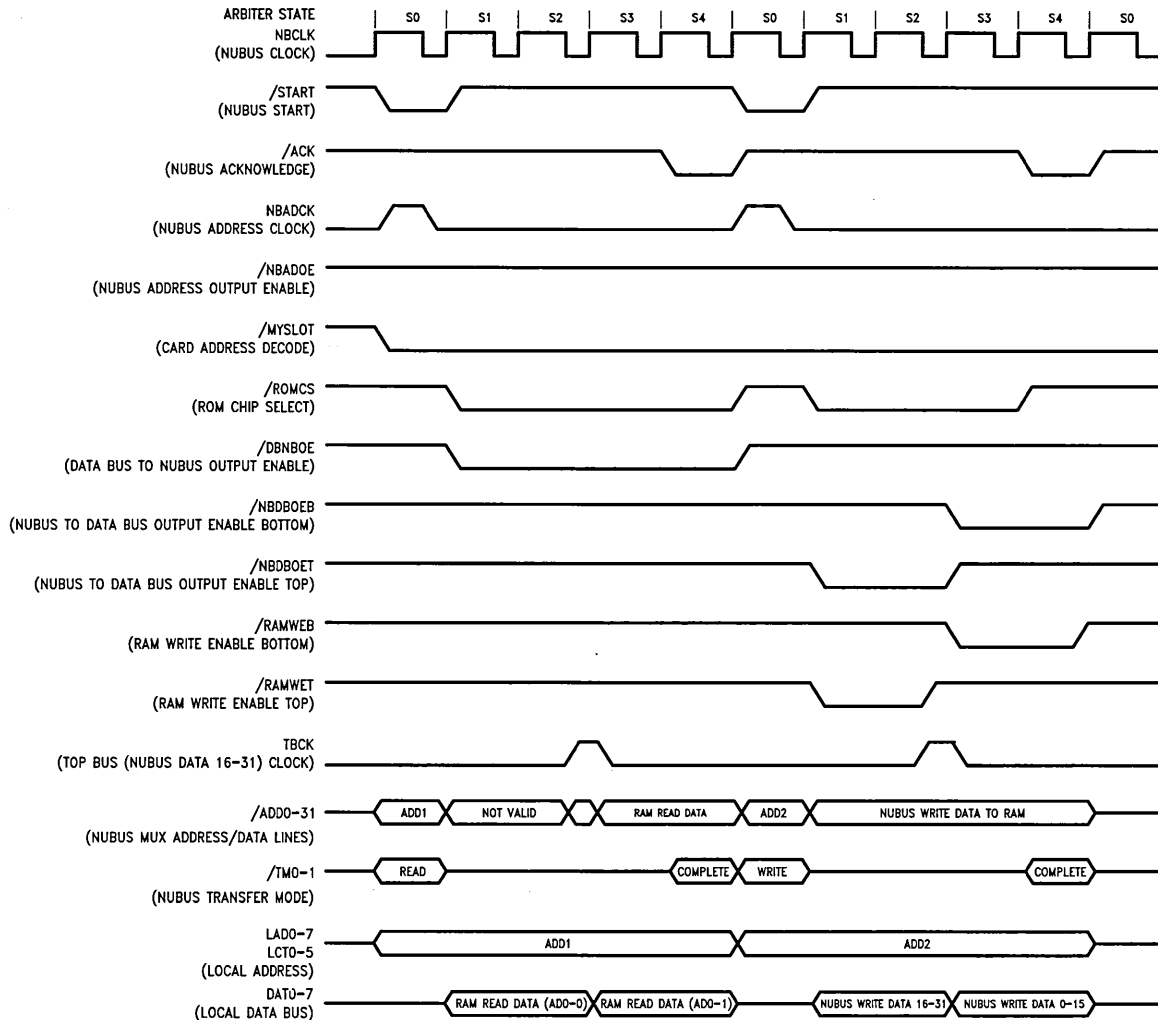


FIGURE 12. NuBus to RAM Read and Write Cycles

TL/F/10805-12

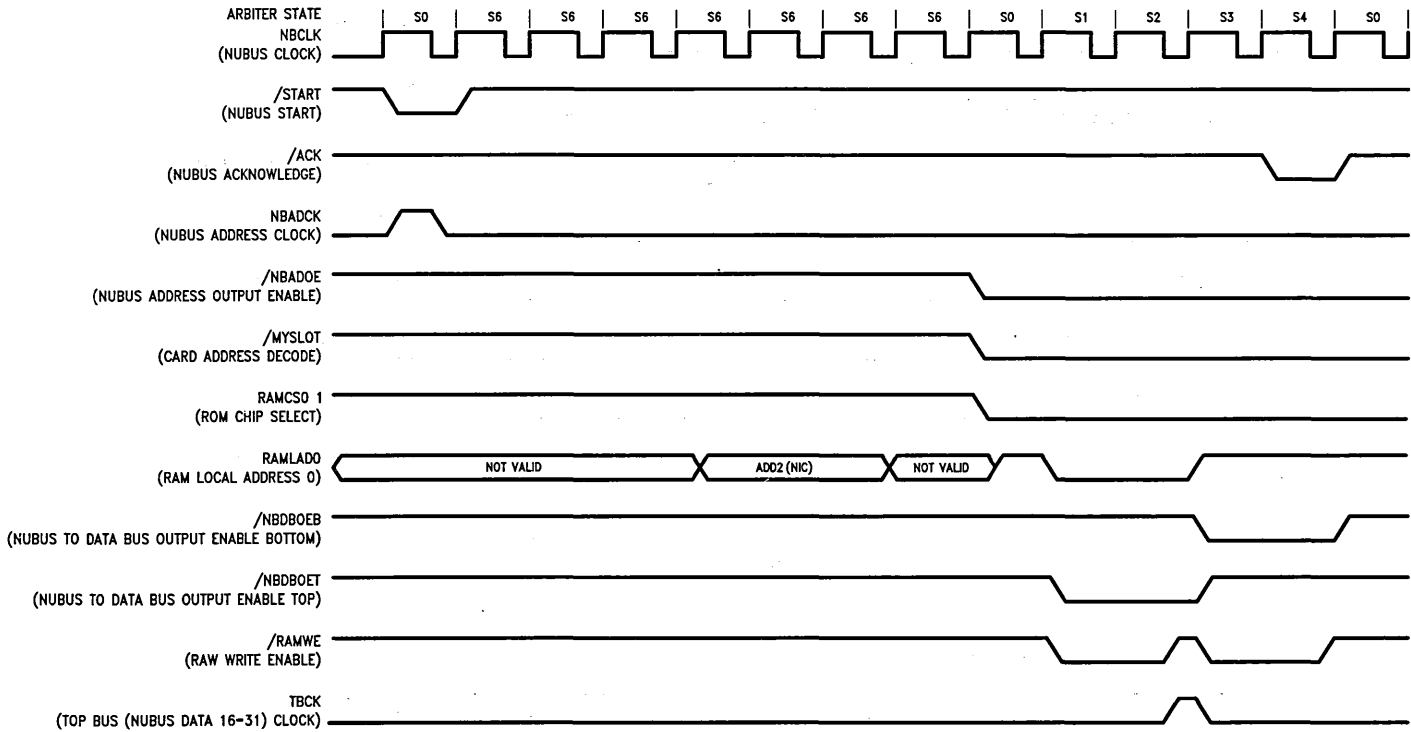


FIGURE 13. NuBus to RAM—Write Cycles with Arbitration



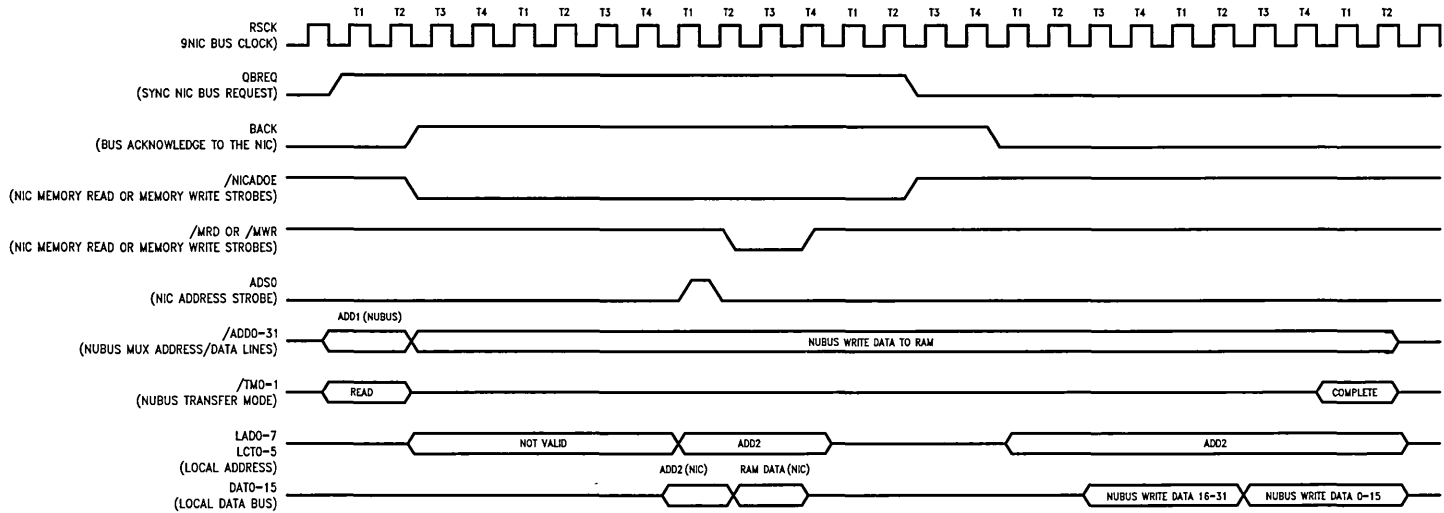


FIGURE 14. NIC to RAM—Write Cycles with Arbitration

TL/F/10805-14

```
module pal_1B

title 'Bus Controller
Andrew Pagnon 7-3-89';

" Modified for using Abel State_Machine language
"32bit Nubus Version"

"declarations"
TRUE = 1;
FALSE = 0;

PAL1B device 'P16R4';

"inputs"
NBCK      pin 1;
QBREQ     pin 2;
QNICACK   pin 3;
MYSLOT    pin 4;
LCT4      pin 5;
BERR      pin 6;
EBSLOT    pin 7;
LCT5      pin 8;
ONE       pin 11;

"outputs"
TM0       pin 12;
TM1       pin 13;
ACK       pin 14;
DASB     pin 15;
BACK     pin 16;
TOP      pin 17;
NBADOE   pin 18;
NICADOE  pin 19;

"Declarations
H,L,CK,XX = 1,0,.C.,.X.;

input = [QBREQ,QNICACK,MYSLOT,LCT4,BERR,EBSLOT,LCT5];

s0 = ^b1111;
s1 = ^b1010;
s2 = ^b1110;
s3 = ^b1011;
s4 = ^b0011;
s5 = ^b0111;
s6 = ^b1101;

equations

enable TM0 = !ACK;
!TM0 = BERR;
```

```
enable TM1 = !ACK;
!TM1 = !ACK;
```

```
enable NBADOE = TRUE;
!NBADOE = BACK & !ACK # BACK & !DASB # BACK & !QBREQ # BACK & !TOP;
```

```
" NBADOE = !BACK* + BREQ . (ACK* . DASB* . BACK* . TOP*)"
"NBADOE IS NOT ACTIVE IF BACK IS ACTIVE OR IF BREQ IS ACTIVE DURING S0"
```

```
enable NICADOE = TRUE;
!NICADOE = QBREQ & !BACK;
```

```
state_diagram [ACK,DASB,BACK,TOP]
```

```
State s0: case (input == [1,XX,XX,XX,XX,XX,XX]) :s6;
           (input == [0,XX, 0,XX,XX, 0,XX]) :s1;
           (input == [0,XX, 1,XX,XX,XX,XX]) :s0; "hold
endcase;
```

```
State s1: case (input == [XX,XX,XX,XX, 0,XX,XX]) :s5; "BERR
           (input == [XX, 0, 0, 0, 1,XX,XX]) :s4; "NIC
           (input == [XX,XX, 0, 1, 1,XX, 1]) :s4; "ROM
           (input == [XX,XX, 0, 1, 1,XX, 0]) :s2; "RAM
           (input == [XX, 1, 0, 0, 1,XX,XX]) :s1; "hold
endcase;
```

```
State s2: goto s3;
```

```
State s3: goto s4;
```

```
State s4: goto s0;
```

```
State s5: goto s0;
```

```
State s6: case (input == [0,XX,XX,XX,XX,XX,XX]) :s0;
           (input == [1,XX,XX,XX,XX,XX,XX]) :s6; "hold
endcase;
```

TL/F/10805-21

## test\_vectors

([NBCK,QBREQ,QNICACK,MYSL0T,LCT4,BERR,EBSL0T,LCT5,ONE] ->  
[TMO,TM1,ACK,DASE,BACK,TOE,NBADOE,NICADOE])

```

[.C.,0,1,1,1,1,1,0] -> [.X.,.X.,.X.,.X.,.X.,.X.,.X.,.X.];
[.C.,0,1,1,1,1,1,0] -> [.X.,.X.,.X.,.X.,.X.,.X.,.X.,.X.];
[.C.,0,1,1,1,1,1,0] -> [.X.,.X.,.X.,.X.,.X.,.X.,.X.,.X.];
[.C.,0,1,1,1,1,1,0] -> [.Z.,.Z.,1,1,1,0,1]; "TEST FOR IDLE S0"
[.C.,1,1,0,0,1,0,0,0] -> [.Z.,.Z.,1,1,0,1,1,0]; "SET NIC BREQ S6"
                                "NB TO NIC START CYCLE "
[.C.,1,1,0,0,1,0,0,0] -> [.Z.,.Z.,1,1,0,1,1,0]; "HOLD NIC BREQ S6"
[.C.,0,1,0,0,1,0,0,0] -> [.Z.,.Z.,1,1,1,1,0,1]; "CLEAR NIC BREQ S0"
[.C.,0,1,0,0,1,0,0,0] -> [.Z.,.Z.,1,0,1,0,0,1]; "NB TO NIC - S1"
[.C.,0,1,0,0,1,0,1,0] -> [.Z.,.Z.,1,0,1,0,0,1]; "WAIT FOR NICACK S1"
[.C.,0,0,0,0,1,0,1,0] -> [ 0 , 0 , 0,0,1,1,0,1]; "NICACK SETS S4"
[.C.,0,1,0,0,1,1,1,0] -> [.Z.,.Z.,1,1,1,1,0,1]; "RETURN TO IDLE S0"
[.C.,0,1,0,0,1,1,1,0] -> [.Z.,.Z.,1,1,1,1,0,1]; "IDLE WITH NO EBSL0T S0"
[.C.,0,1,0,1,1,0,0,0] -> [.Z.,.Z.,1,0,1,0,0,1]; "NB TO RAM - S1"
[.C.,0,1,0,1,0,0,0,0] -> [ 1 , 0 , 0,1,1,1,0,1]; "BERR SETS S5"
[.C.,1,1,0,1,0,1,1,0] -> [.Z.,.Z.,1,1,1,1,1,1]; "IDLE NIC BREQ SETS S0"
[.C.,1,1,0,1,0,1,1,0] -> [.Z.,.Z.,1,1,0,1,1,0]; "NIC MASTER S6"
[.C.,1,1,0,1,1,0,1,0] -> [.Z.,.Z.,1,1,0,1,1,0]; "NB TO MEM -NIC MASTER S6"
[.C.,0,1,0,1,1,0,1,0] -> [.Z.,.Z.,1,1,1,1,0,1]; "RETURN TO IDLE S0"
[.C.,0,1,0,1,1,0,1,0] -> [.Z.,.Z.,1,0,1,0,0,1]; "NB TO ROM - S1"
[.C.,1,1,0,1,1,0,1,0] -> [ 0 , 0 , 0,0,1,1,0,1]; "NIC BREQ - SET NB ACK S4"
[.C.,1,1,0,1,1,1,1,0] -> [.Z.,.Z.,1,1,1,1,1,1]; "RETURN TO IDLE S0"
[.C.,1,1,1,1,1,1,1,0] -> [.Z.,.Z.,1,1,0,1,1,0]; "NIC MASTER S6"
[.C.,0,1,1,1,1,1,1,0] -> [.Z.,.Z.,1,1,1,1,0,1]; "RETURN TO IDLE S0"
[.C.,0,1,0,1,1,0,0,0] -> [.Z.,.Z.,1,0,1,0,0,1]; "NB TO RAM S1"
[.C.,1,1,0,1,1,0,0,0] -> [.Z.,.Z.,1,1,1,0,0,1]; "NIC BREQ-LTCH DA16-31 S2"
[.C.,1,1,0,1,1,0,0,0] -> [.Z.,.Z.,1,0,1,1,0,1]; "EB DAO-15 S3"
[.C.,1,1,0,1,1,0,0,0] -> [ 0 , 0 , 0,0,1,1,0,1]; "ACK to NUBUS S4"
[.C.,1,1,0,1,1,0,0,0] -> [.Z.,.Z.,1,1,1,1,1,1]; "RETURN TO IDLE S0"
[.C.,1,1,1,1,1,1,1,0] -> [.Z.,.Z.,1,1,0,1,1,0]; "NIC MASTER S6"
[.C.,0,1,1,1,1,1,1,0] -> [.Z.,.Z.,1,1,1,1,0,1]; "RETURN TO IDLE S0"
[.C.,0,1,1,1,1,1,1,0] -> [.Z.,.Z.,1,1,1,1,0,1]; "STAY IN IDLE S0"

```

end pal\_1B

TL/F/10805-22

```

module pal_2

title 'Memory decoder
Andrew Pagnon 8-3-89';

"declarations"
TRUE = 1;
FALSE = 0;

PAL2 device 'P16L8';

"inputs"
LCT0      pin 1;
LCT1      pin 2;
LCT2      pin 3;
LCT3      pin 4;
LCT4      pin 5;
LCT5      pin 6;
MYSLOT    pin 7;
DASB      pin 8;
NICADOE   pin 9;
MSRAMSL   pin 11;

"outputs"
RAMCS1     pin 12;
ROMCS      pin 13;
NICCS      pin 14;
RAMCS0     pin 15;
RAMCS3     pin 16;
RAMCS2     pin 17;
BERR       pin 18;
SRD        pin 19;

equations

enable RAMCS0 = TRUE;

!RAMCS0 = !NICADOE & !MSRAMSL
# LCT0 & LCT1 & !LCT2 & !LCT3 & LCT4 & !LCT5 & !MYSLOT " Write byte 0 "
# !LCT0 & LCT1 & LCT2 & !LCT3 & LCT4 & !LCT5 & !MYSLOT " Write hw 0 "
# !LCT0 & LCT1 & !LCT2 & !LCT3 & LCT4 & !LCT5 & !MYSLOT " Write word "
# !LCT1 & LCT4 & !LCT5 & !MYSLOT; " Read "

enable RAMCS1 = TRUE;

!RAMCS1 = !NICADOE & !MSRAMSL
# LCT0 & LCT1 & LCT2 & !LCT3 & LCT4 & !LCT5 & !MYSLOT " Write byte 1 "
# !LCT0 & LCT1 & LCT2 & !LCT3 & LCT4 & !LCT5 & !MYSLOT " Write hw 0 "
# !LCT0 & LCT1 & !LCT2 & !LCT3 & LCT4 & !LCT5 & !MYSLOT " Write word "
# !LCT1 & LCT4 & !LCT5 & !MYSLOT; " Read "

```

TL/F/10805-23

```

enable RAMCS2 = TRUE;

!RAMCS2 = !NICADOE & MSRAMSL
# LCT0 & LCT1 & !LCT2 & LCT3 & LCT4 & !LCT5 & !MYSLOT " Write byte 2"
# !LCT0 & LCT1 & LCT2 & LCT3 & LCT4 & !LCT5 & !MYSLOT " Write hw 1 "
# !LCT0 & LCT1 & !LCT2 & !LCT3 & LCT4 & !LCT5 & !MYSLOT " Write word "
# !LCT1 & LCT4 & !LCT5 & !MYSLOT; " Read "

enable RAMCS3 = TRUE;

!RAMCS3 = !NICADOE & MSRAMSL
# LCT0 & LCT1 & LCT2 & LCT3 & LCT4 & !LCT5 & !MYSLOT " Write byte 3 "
# !LCT0 & LCT1 & LCT2 & LCT3 & LCT4 & !LCT5 & !MYSLOT " Write hw 1 "
# !LCT0 & LCT1 & !LCT2 & !LCT3 & LCT4 & !LCT5 & !MYSLOT " Write word "
# !LCT1 & LCT4 & !LCT5 & !MYSLOT; " Read "

enable ROMCS = TRUE;
!ROMCS = !LCT1 & LCT4 & LCT5 & !MYSLOT & !DASB; " Read "

enable NICCS = TRUE;
!NICCS = !DASB & !MYSLOT & !LCT4 & LCT5; " NIC Register Read or Write "

enable SRD = TRUE;
!SRD = !NICCS & !LCT1; " NIC register read "

enable BERR = TRUE;
!BERR = LCT1 & LCT4 & LCT5 & !MYSLOT "ROM WRITE"
# !LCT4 & !LCT5 & !MYSLOT; "NOT IN CARD"

test_vectors
([LCT0,LCT1,LCT2,LCT3,LCT4,LCT5,MYSLOT,DASB,NICADOE,MSRAMSL] ->
[RAMCS0, RAMCS1, RAMCS2, RAMCS3, ROMCS, NICCS, BERR, SRD])
[.X.,.X.,.X.,.X.,.X.,.X.,1,.X.,0,0] -> [0,0,1,1,1,1,1,1,1];"NIC RD/WT RAM HW0"
[.X.,.X.,.X.,.X.,.X.,.X.,1,.X.,0,1] -> [1,1,0,0,1,1,1,1,1];"NIC RD/WT RAM HW1"
[1,1,0,0,0,1,0,0,1,.X.] -> [1,1,1,1,1,0,1,1,1];"NB WT NIC BYTE0"
[0,0,0,0,0,1,0,0,1,.X.] -> [1,1,1,1,1,0,1,0,1];"NB RD NIC WD"
[0,0,0,0,1,1,0,.X.,1,.X.] -> [1,1,1,1,0,1,1,1,1];"NB RD ROM WD"
[0,0,0,0,1,0,0,.X.,1,.X.] -> [0,0,0,0,1,1,1,1,1];"NB RD RAM WD"
[1,1,0,0,1,0,0,.X.,1,.X.] -> [0,1,1,1,1,1,1,1,1];"NB WT RAM BYTE0"
[1,1,1,0,1,0,0,.X.,1,.X.] -> [1,0,1,1,1,1,1,1,1];"NB WT RAM BYTE1"
[1,1,0,1,1,0,0,.X.,1,.X.] -> [1,1,0,1,1,1,1,1,1];"NB WT RAM BYTE2"
[1,1,1,1,1,0,0,.X.,1,.X.] -> [1,1,1,0,1,1,1,1,1];"NB WT RAM BYTE3"
[0,1,1,0,1,0,0,.X.,1,.X.] -> [0,0,1,1,1,1,1,1,1];"NB WT RAM HW0"
[0,1,1,1,1,0,0,.X.,1,.X.] -> [1,1,0,0,1,1,1,1,1];"NB WT RAM HW1"
[0,1,0,0,1,0,0,.X.,1,.X.] -> [0,0,0,0,1,1,1,1,1];"NB WT RAM WD"
[1,0,.X.,.X.,1,0,0,.X.,1,.X.] -> [0,0,0,0,1,1,1,1,1];"NB RD RAM BY BERR"
[0,0,.X.,1,1,0,0,.X.,1,.X.] -> [0,0,0,0,1,1,1,1,1];"NB RD RAM H1,BL BERR"
[0,0,1,0,1,0,0,.X.,1,.X.] -> [0,0,0,0,1,1,1,1,1];"NB RD RAM H0 BERR"
[.X.,1,.X.,.X.,1,1,0,.X.,1,.X.] -> [1,1,1,1,1,1,0,1,1];"NB WT ROM BERR"
[.X.,.X.,.X.,.X.,0,0,0,.X.,1,.X.] -> [1,1,1,1,1,1,0,1,1];"ADD NOT IN CARD BERR"

```

```

end pal_2

```

TL/F/10805-24

```

module pal_3

title 'Memory and buffer control
Andrew Pagnon 23-3-89';

"declarations"
TRUE = 1;
FALSE = 0;

PAL3 device 'P16L8';

"inputs"
START      pin 1;
NBCLK      pin 2;
ACK        pin 3;
DASB       pin 4;
MYSLOT     pin 5;
LCT1       pin 6;
MRD        pin 7;
TOP        pin 8;
LCT5       pin 9;
RESET      pin 11;

"outputs"
DBNBOE     pin 12;
NBADCK     pin 13;
C          pin 14;
RAMWEB     pin 15;
NBDBOEB    pin 16;
A          pin 17;
EBSLOT     pin 18;
ACKN       pin 19;

equations

enable A = TRUE;
!A = A & !NBADCK & START & NBCLK # !A & !NBADCK & START
    # !A & !NBADCK & !NBCLK # !A & !NBADCK & !START & NBCLK;

enable NBADCK = TRUE;
!NBADCK = A & NBADCK & !START & !NBCLK # A & !NBADCK & !START
    # A & !NBADCK & !NBCLK # A & !NBADCK & START & NBCLK
    # !A & !NBADCK & START # !A & !NBADCK & !NBCLK;

enable C = TRUE;
!C = !C & RAMWEB # !C & !RAMWEB & ACK & !DASB
    # C & RAMWEB & NBDBOEB;

enable RAMWEB = TRUE;
!RAMWEB = !C & RAMWEB & !NBDBOEB # !C & !RAMWEB & ACK & !DASB
    # !C & !RAMWEB & !ACK # !C & !RAMWEB & DASB
    # C & !RAMWEB & NBCLK;

```

TL/F/10805-25

```

enable NBDBOEB = TRUE;
!NBDBOEB = LCT1 & !DASB & !MYSLOT & TOP & !LCT5;

enable DENBOE = TRUE;
!DENBOE = !LCT1 & !DASB & !MYSLOT
# !LCT1 & !TOP & !MYSLOT;

enable EBSLOT = TRUE;
!EBSLOT = EBSLOT & !START & RESET # !EBSLOT & ACK & RESET;

enable ACKN = !ACK;
!ACKN = !ACK;

```

## test\_vectors

```

([START,NBCLK,ACK,DASB,MYSLOT,LCT1, TOP,LCT5,RESET] ->
[A,NBADCK,C, RAMWEB,NBDBOEB, DENBOE, EBSLOT, ACKN])
[1,1,1,1,1,.X.,1,.X.,0] -> [.X.,.X.,.X.,.X.,.X.,.X.,.X.,.X.];
[1,0,1,1,1,.X.,1,.X.,1] -> [.X.,.X.,.X.,.X.,.X.,.X.,.X.,.X.];
[1,1,1,1,1,.X.,1,.X.,1] -> [.X.,.X.,.X.,.X.,1,1,1,.Z.];
[1,0,1,1,1,.X.,1,.X.,1] -> [.X.,.X.,.X.,.X.,1,1,1,.Z.];
[0,1,1,1,1,.X.,1,.X.,1] -> [.X.,1,0,1,1,1,0,.Z.]; "S0,START LOW NBCKL HIGH"
[0,1,1,1,0,1,1,0,1] -> [1,1,0,1,1,1,0,.Z.]; "S0,MYSLOT* LOW,S0"
[0,0,1,1,0,1,1,0,1] -> [1,0,0,1,1,1,0,.Z.]; "S0,NBCLK1=0,S0"
[1,1,1,0,0,1,0,0,1] -> [0,0,0,1,1,1,0,.Z.]; "S1,NBCLK2=1,START*=1"
[1,1,1,0,0,1,0,0,1] -> [0,0,0,1,1,1,0,.Z.]; "S1"
[1,0,1,0,0,1,0,0,1] -> [0,0,0,1,1,1,0,.Z.]; "S1,NBCLK2=0"
[1,1,1,0,1,0,1,0,1] -> [0,0,0,1,1,1,0,.Z.]; "S2 NBCLK3=1"
[1,1,1,1,0,1,0,0,1] -> [0,0,0,1,1,1,0,.Z.]; "S2"
[1,1,1,1,0,1,0,0,1] -> [0,0,0,1,1,1,0,.Z.]; "S2"
[1,0,1,1,0,1,0,0,1] -> [0,0,0,1,1,1,0,.Z.]; "S2,NBCLK3=0"
[1,1,1,0,0,1,1,0,1] -> [0,0,0,0,0,1,0,.Z.]; "S3,NBCLK4=1"
[1,1,1,0,0,1,1,0,1] -> [0,0,0,0,0,1,0,.Z.]; "S3"
[1,0,1,0,0,1,1,0,1] -> [0,0,0,0,0,1,0,.Z.]; "S3,NBCLK=0"
[1,1,0,0,0,1,1,0,1] -> [0,0,1,0,0,1,1,0]; "S4"
[1,0,0,0,0,1,1,0,1] -> [0,0,1,1,0,1,1,0]; "S4"
[1,1,1,1,0,1,1,0,1] -> [0,0,0,1,1,1,1,.Z.]; "S0"
[1,0,1,1,0,1,1,0,1] -> [0,0,0,1,1,1,1,.Z.]; "S0"
[0,1,1,1,0,1,1,1,1] -> [1,1,0,1,1,1,0,.Z.]; "S0,START*=0,NBCLK5=1"
[0,1,1,1,0,1,1,1,1] -> [1,1,0,1,1,1,0,.Z.]; "S0,NIC OR ROM WRITE"
[0,0,1,1,0,1,1,1,1] -> [1,0,0,1,1,1,0,.Z.]; "S0,NBCLK5=0"
[0,0,1,1,0,1,1,1,1] -> [1,0,0,1,1,1,0,.Z.]; "S0,MYSLOT*=1"
[1,1,1,0,0,1,0,1,1] -> [0,0,0,1,1,1,0,.Z.]; "S1,NBCLK6=1"
[1,1,1,0,0,1,0,1,1] -> [0,0,0,1,1,1,0,.Z.]; "S1"
[1,1,0,0,0,1,1,1,1] -> [0,0,0,1,1,1,1,0]; "S4"
[1,0,0,0,0,1,1,1,1] -> [0,0,0,1,1,1,1,0]; "S4"
[1,1,1,1,0,1,1,1,1] -> [0,0,0,1,1,1,1,.Z.]; "S0"
[1,0,1,1,.X.,.X.,1,.X.,1]->[0,0,0,1,1,1,1,.Z.]; "S0"

```

end pal\_3



```

module pal_4

title '32 bit Nubus control
Andrew Pagnon 11-10-89';

"declarations"
TRUE = 1;
FALSE = 0;

PAL6 device 'P16L8';

"inputs"
TOP      pin 1;
NBCLK    pin 2;
DASB     pin 3;
BACK     pin 4;
LAD0     pin 5;
MYSLOT   pin 6;
ACK      pin 7;
BERR     pin 8;
LCT1     pin 9;
LCT4     pin 11;
LCT5     pin 18;

"outputs"
RAMLAD0  pin 12;
TBCK     pin 13;
NBDBOET  pin 14;
RAMWET   pin 15;
D        pin 16;
DBNBOEB  pin 17;

equations

enable RAMLAD0 = TRUE;
!RAMLAD0 = !BACK & !LAD0 # BACK & !TOP;

enable TBCK = TRUE;
!TBCK = TOP # NBCLK # !DASB;

enable NBDBOET = TRUE;
!NBDBOET = LCT1 & !MYSLOT & !TOP "RAM"
# LCT1 & !MYSLOT & !DASB & BERR & !LCT4; "NIC"

enable RAMWET = TRUE;
!RAMWET = !D & RAMWET & !NBDBOET
# !D & !RAMWET & !DASB & !MYSLOT & ACK
# !D & !RAMWET & DASB
# !D & !RAMWET & !ACK
# !D & !RAMWET & MYSLOT
# D & !RAMWET & NBCLK;

enable D = TRUE;
!D = !D & RAMWET
# !D & !RAMWET & !DASB & ACK & !MYSLOT
# D & RAMWET & NBDBOET;

```

TL/F/10805-27

```

enable DBNBOEB = TRUE;
!DBNBOEB = !LCT1 & !DASB & !MYSLOT & !LCT5
# !LCT1 & !TOP & !MYSLOT & !LCT5;

```

## test\_vectors

```

([TOP,NBCLK,DASB,BACK,LAD0,MYSLOT,ACK,BERR,LCT1,LCT4,LCT5] ->
[RAMLAD0,TBCK,NBDOET,RAWWET,DBNBOEB])
[1,1,1,0,1,1,1,1,1,X.,X.,X.] -> [X.,X.,X.,X.,X.];"RESET"
[1,0,1,0,1,1,1,1,1,X.,X.,X.] -> [X.,X.,X.,X.,X.];"RESET"
[1,1,1,0,1,1,1,1,1,X.,X.,X.] -> [1,0,1,1,1];"LAD0=1,BACK*=0"
[1,0,1,0,1,1,1,1,1,X.,X.,X.] -> [1,0,1,1,1];
[1,1,1,0,0,1,1,1,1,X.,X.,X.] -> [0,0,1,1,1];"LAD0=0,BACK*=0"
[1,0,1,0,0,1,1,1,1,X.,X.,X.] -> [0,0,1,1,1];
[1,1,1,1,0,1,1,1,1,X.,X.,X.] -> [0,0,1,1,1];"LAD0=0,BACK*=1, TOP=1->RAMLAD0=1"
[1,0,1,1,0,1,1,1,1,X.,X.,X.] -> [0,0,1,1,1];
[0,1,0,1,0,0,1,1,0,1] -> [1,0,0,0,1];"S1,NIC WT"
[0,0,0,1,X.,0,1,1,0,1] -> [1,0,0,0,1];"S1,NIC WT,NBCLK=0"
[1,1,0,1,X.,0,0,1,1,0,1] -> [0,0,0,0,1];"S4,NIC WT,ACK=0"
[1,0,0,1,X.,0,0,1,1,0,1] -> [0,0,0,1,1];"S4,NIC WT,ACK=0,NBCLK=0"
[1,1,1,1,X.,0,1,1,0,1] -> [0,0,1,1,1];"S0"
[1,0,1,1,X.,0,1,1,0,1,1] -> [0,0,1,1,1];"S0"
[0,1,0,1,X.,0,1,1,0,1,1] -> [1,0,1,1,1];"S1,ROM RD"
[0,0,0,1,X.,0,1,1,0,1,1] -> [1,0,1,1,1];"S1,ROM RD,NBCLK=0"
[1,1,0,1,X.,0,0,1,0,1,1] -> [0,0,1,1,1];"S4,ROM RD,ACK=0"
[1,0,0,1,X.,0,0,1,0,1,1] -> [0,0,1,1,1];"S4,ROM RD,ACK=0,NBCLK=0"
[1,1,1,1,X.,0,1,1,0,1,1] -> [0,0,1,1,1];"S0"
[1,0,1,1,X.,0,1,1,0,1,0] -> [0,0,1,1,1];"S0"
[0,1,0,1,X.,0,1,1,0,1,0] -> [1,0,1,1,0];"S1,RAM RD"
[0,0,0,1,X.,0,1,1,0,1,0] -> [1,0,1,1,0];"S1,RAM RD,NBCLK=0"
[0,1,1,1,X.,0,1,X.,0,1,0] -> [1,0,1,1,0];"S2,RAM RD"
[0,0,1,1,X.,0,1,X.,0,1,0] -> [1,1,1,1,0];"S2,RAM RD,NBCLK=0,TBCK=1"
[1,1,0,1,X.,0,1,X.,0,1,0] -> [0,0,1,1,0];"S3,RAM RD, TOP=1"
[1,0,0,1,X.,0,1,X.,0,1,0] -> [0,0,1,1,0];"S3,RAM RD,NBCLK=0"
[1,1,0,1,X.,0,1,X.,0,1,0] -> [0,0,1,1,0];"S4,RAM RD,ACK=0"
[1,0,0,1,X.,0,0,X.,0,1,0] -> [0,0,1,1,0];"S4,RAM RD,ACK=0,NBCLK=0"
[1,1,1,1,X.,0,1,X.,0,1,0] -> [0,0,1,1,1];"S0"
[1,0,1,1,X.,0,1,1,1,1,0] -> [0,0,1,1,1];"S0"
[0,1,0,1,X.,0,1,1,1,1,0] -> [1,0,0,0,1];"S1,RAM WT"
[0,0,0,1,X.,0,1,1,1,1,0] -> [1,0,0,0,1];"S1,RAM WT,NBCLK=0"
[1,1,1,1,X.,0,0,X.,1,1,0] -> [0,0,1,0,1];"S5,RAM WT,BERR=0"
[1,0,1,1,X.,0,0,X.,1,1,0] -> [0,0,1,1,1];"S5,RAM WT,BERR=0,NBCLK=0"
[1,1,1,1,X.,0,1,X.,1,1,0] -> [0,0,1,1,1];"S0"
[1,0,1,1,X.,0,1,1,1,1,1] -> [0,0,1,1,1];"S0"
[0,1,0,1,X.,0,1,1,1,1,1] -> [1,0,0,0,1];"S1,ROM WT"
[0,0,0,1,X.,0,1,1,1,1,1] -> [1,0,0,0,1];"S1,ROM WT,NBCLK=0"
[1,1,1,1,X.,0,0,X.,1,1,1] -> [0,0,1,0,1];"S5,ROM WT,BERR=0"
[1,0,1,1,X.,0,0,X.,1,1,1] -> [0,0,1,1,1];"S5,ROM WT,BERR=0,NBCLK=0"
[1,1,1,1,X.,0,1,X.,1,1,1] -> [0,0,1,1,1];"S0"
[1,0,1,1,X.,0,1,1,1,1,0] -> [0,0,1,1,1];"S0"
[0,1,0,1,X.,0,1,1,1,1,0] -> [1,0,0,0,1];"S1,RAM WT"
[0,0,0,1,X.,0,1,1,1,1,0] -> [1,0,0,0,1];"S1,RAM WT,NBCLK=0"
[0,1,1,1,X.,0,1,X.,1,1,0] -> [1,0,0,0,1];"S2,RAM WT"
[0,0,1,1,X.,0,1,X.,1,1,0] -> [1,1,0,1,1];"S2,RAM WT,NBCLK=0,TBCK=1"
[1,1,0,1,X.,0,1,X.,1,1,0] -> [0,0,1,1,1];"S3,RAM WT, TOP=1"
[1,0,0,1,X.,0,1,X.,1,1,0] -> [0,0,1,1,1];"S3,RAM WT,NBCLK=0"

```

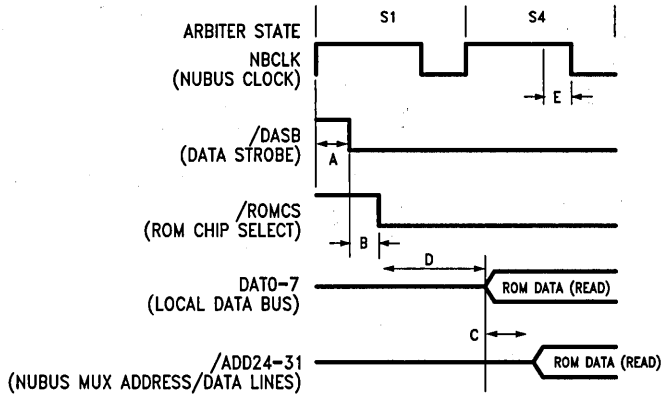
```

[1,1,0,1,.X.,0,1,.X.,1,1,0] -> [0,0,1,1,1];"S4, RAM WT, ACK=0"
[1,0,0,1,.X.,0,0,.X.,1,1,0] -> [0,0,1,1,1];"S4, RAM WT, ACK=0, NBCLK=0"
[1,1,1,1,.X.,0,1,.X.,1,1,0] -> [0,0,1,1,1];"S0"
[1,0,1,1,.X.,0,1,.X.,0,1,1] -> [0,0,1,1,1];"S0"
[0,1,0,1,.X.,0,1,.X.,0,1,1] -> [1,0,1,1,1];"S1, ROM RD"
[0,0,0,1,.X.,0,1,1,0,1,1] -> [1,0,1,1,1];"S1, ROM RD, NBCLK=0"
[1,1,0,1,.X.,0,0,.X.,0,1,1] -> [0,0,1,1,1];"S4, ROM RD, ACK=0"
[1,0,0,1,.X.,0,0,.X.,0,1,1] -> [0,0,1,1,1];"S4, ROM RD, ACK=0, NBCLK=0"
[1,1,1,1,.X.,0,1,.X.,0,1,1] -> [0,0,1,1,1];"S0"
[1,0,1,1,.X.,1,1,.X.,.X.,.X.,.X.] -> [1,0,1,1,1];"S0"
[1,1,1,1,.X.,1,1,.X.,.X.,.X.,.X.] -> [1,0,1,1,1];"S0"
[.X.,.X.,.X.,0,0,.X.,.X.,.X.,.X.,.X.,.X.] ->
[0,.X.,.X.,.X.,.X.,.X.];"BACK*=0, LAD&RLAD=0"
[.X.,.X.,.X.,0,1,.X.,.X.,.X.,.X.,.X.,.X.] ->
[1,.X.,.X.,.X.,.X.,.X.];"BACK*=0, LAD&RLAD=1"

```

end pal\_6

TL/F/10805-29



TL/F/10805-15

FIGURE A-1. Detailed NuBus to ROM Read Timing

**TIMINGS**

- A = 15 ns, 16R4B prop delay, NbClk to DaSb
- B = 15 ns, 16L8B prop delay, DaSb to ROMCS
- C = 8 ns, F651 prop delay, Data Bus to NuBus
- D = 35 ns, Max ROM enable access time, ROMCS to Data Bus
- E = 21 ns, NuBus data set up time, NuBus data to NbClk low (S4)

**Note:** The address to the ROM is valid midway through the S0 state. Therefore the ROM address access time (70 ns) is not in the timing critical path.

**ROM READ DATA SET UP TIME**

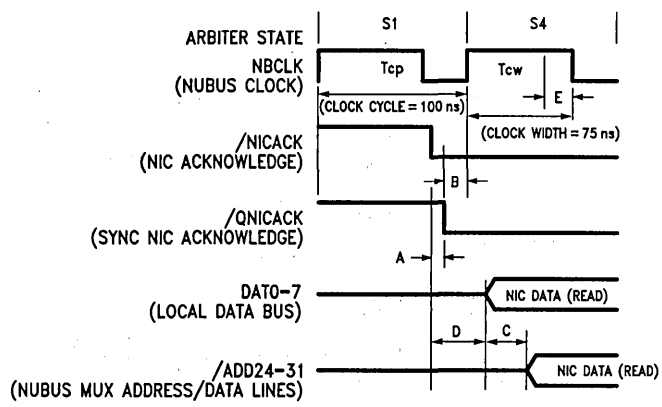
**Spec Times To Be Met**

Read data set up time to NuBus  
 (NbClk(S4)low) = 21 ns  
 (Tsu in Nubus spec)

**Adapter Card Times**

Read data set up time  
 $= T_{cp}(S1) + T_{cw}(S4) - A - B - C - D$   
 $= 100 + 75 + 15 + 15 - 8 - 35$   
 $= 102 \text{ ns (81 ns to spare)}$

Data hold times are as per RAM read cycles



TL/F/10805-16

FIGURE A-2. Detailed NuBus to DP8390 Read Timing

**TIMINGS**

- A = 4 ns, Min F175 prop delay, NICACK to QNICACK
  - B = 10 ns, Typ 16R4B Set Up, QNICACK to NBCLK
  - C = 8 ns, F651 prop delay, Data Bus to NuBus
  - D = 55 ns, NIC Register access time, NICACK to Data Bus
  - E = 21 ns, NuBus Data set up time, NuBus data to NbCik
- Data hold times are as per RAM read cycles

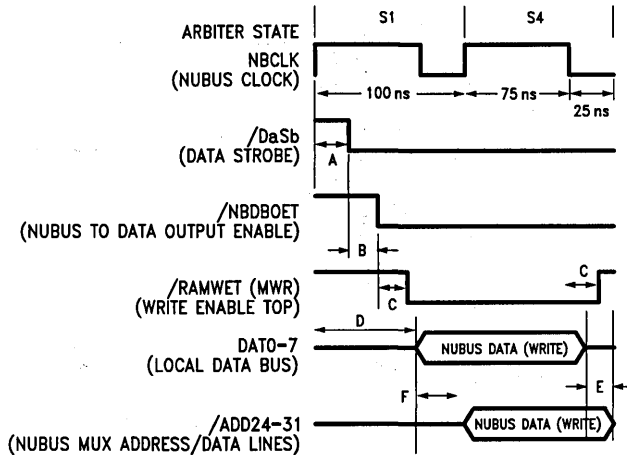
**NIC REGISTER READ DATA SET UP TIME**

**Spec Times To Be Met**

Read data set up time to NuBus(NbCik(S4)low) = 21 ns  
(Tsu in NuBus spec)

**Adapter Card Times**

Read data set up time =  
 $T_{cw}(S2) + A + B - C - D = 75 + 4 + 10 - 8 - 55$   
 = 26 ns (5 ns to spare)  
 (Note B is a typical value)



TL/F/10805-17

FIGURE A-3. NuBus to NIC Write Timing

**TIMINGS**

$A + B + C = 45$  ns, 16R4B and 16L8B delay,  
NbClk to RAMWET

$D = 52$  ns, NuBus data enable ( $T_{on} + 2T_{pd}$ ), DaSb to  
ROMCS

$E = 8$  ns, F651 prop delay, Data Bus to NuBus

$F = 25$  ns, Nubus data hold time ( $T_{cp} - T_{cw}$ )

$G = 2$  ns, F651 min prop delay, Local bus to NuBus

For  $T_{cp}$ ,  $T_{cw}$ ,  $T_{on}$ ,  $T_{pd}$  see RAM read timings

**NIC REGISTER WRITE DATA SET UP TIME****Spec Times To Be Met**

Write data set up time to end of MWR = 20 ns (NIC spec)

**Adapter Card Times**

Write data set up time to end of MWR =

$T_{cp}(S1) + T_{cw}(S2) - D - E = 115$  ns (95 ns to spare).

**NIC REGISTER WRITE DATA HOLD TIME****Spec Times To Be Met**

Write data hold time after MWR = 17 ns (NIC spec)

**Adapter Card Times**

Write data hold time after MWR =  $F + E - C = 17$  ns  
( $C = 10$  ns (D series PAL))

**NIC REGISTER WRITE WIDTH FROM ACK****Spec Times To Be Met**

Write width from ACK = 50 ns min (NIC spec)

**Adapter Card Times**

As RAMWET (MWR) is set before S4,

ACK to RAMWET > 75 ns (> 25 ns to spare).

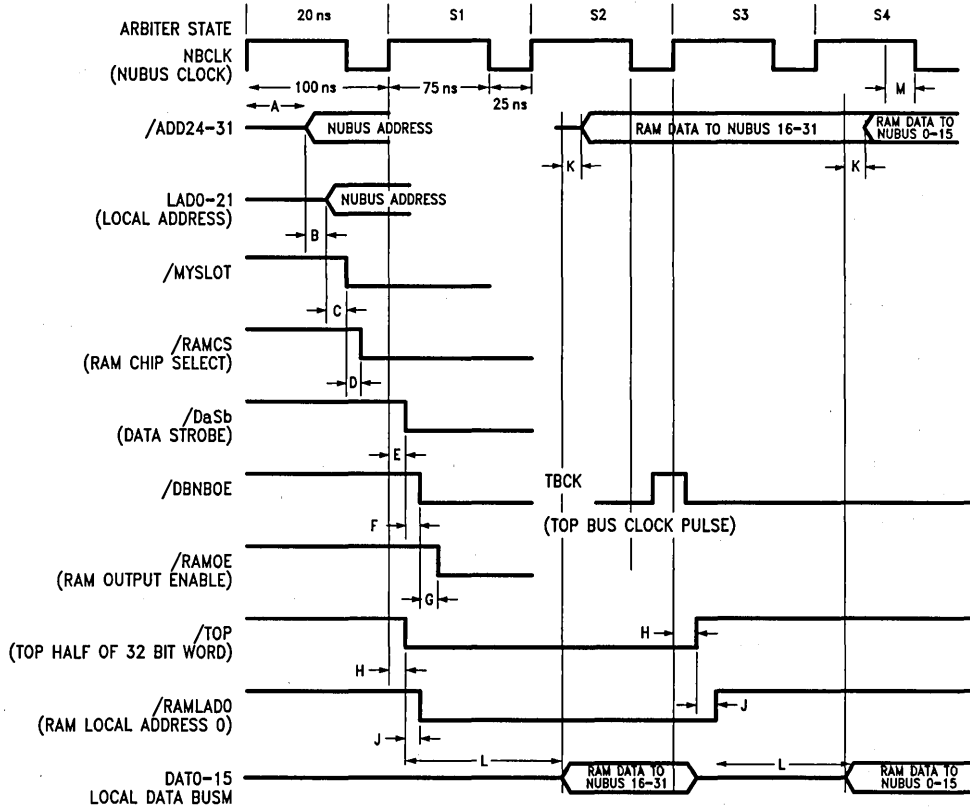


FIGURE A-4. NuBus to RAM Read Timing and Address Recognition Timing

TL/F/10805-18

K = 8 ns, F651 prop delay, Local data bus to NuBus  
 P = 15 ns, PAL16L8B prop delay, NbClk(S2) low to TBCK

**Spec Times To Be Met**

RAM Read data set up time to NuBus  
 (NbClk(S4)low) =  
 M = 21 ns (Tsu in NuBus spec)

RAM Read data set up time in TBCK = 7 ns  
 (F651 spec)

TBCK minimum pulse width = 7 ns  
 (F651 spec)

**Adapter Card Times**

NbClk(S1) to RAM Read data (bottom 16 bits) on local data bus = H+J+L = 130 ns

Set up time to TBCK =  $Tcp(S1+S2) - 130 = 70$  ns  
 (63 ns to spare)

NbClk(S3) to RAM Read data (32 bits) on NuBus = H+J+L+K = 138 ns

**Adapter Card Times**

Data buffer enabling signals are not cleared until after NbClk(S3) goes high

Hold time on Local data bus after TBCK >  $Tcp(S2) - Tcw(S2) - P = 10$  ns  
 (> 10 ns to spare)

Data must be held on NuBus while NbClk(S4) is low and as buffer enabling signals are not cleared until the next rising edge of NbClk, NuBus hold time is met

**Address Recognition**

= 52 ns, Nubus address valid (Ton+2Tpd), NbClk to NuBus

= 10 ns, F533 prop delay, Nubus to Local Address

= 11 ns, F521 prop delay, Local address to MySlot

**TIMINGS**

A = 15 ns, B series PAL prop delay, NbClk to DaSb or TOP  
 B = 15 ns, PAL16L8B prop delay, DaSb or TOP to NBDBOE

C = 15 ns, PAL16L8B prop delay, NBDBOE to RAMWET/B

D = 6.5 ns, F11 prop delay, RAMWET/B to RAMWE

E = 52 ns, NuBus data turn on time, NbClk(S1) to Local data bus

F = 8 ns, F651 prop delay, Local data bus to NuBus

**RAM CE TO END OF WE**

**Spec Times To Be Met**

RAM CE to end of RAMWE = 85 ns (RAM spec)

**Adapter Card Times**

RAMCE is driven during S0 (See RAM read timing)

RAMCE to end of RAMWE >  $Tcp(S1) + Tcw(S2) = 175$  ns (>90 ns to spare)

**RAM WE PULSE WIDTH**

**Spec Times To Be Met**

RAMWE pulse width = 70 ns min (RAM spec)

**Adapter Card Times**

RAMWE pulse width >  $Tcp(S1) + Tcw(S2) - A - B - C - D = 123.5$  ns (53.5 ns to spare)

**DATA VALID TO END OF WE**

**Spec Times To Be Met**

Data valid to end of RAMWE = 50 ns min (RAM spec)

**Adapter Card Times**

Data valid to end of RAMWE =  $Tcp(S1) + Tcw(S2) - E - F = 115$  ns (65 ns to spare)

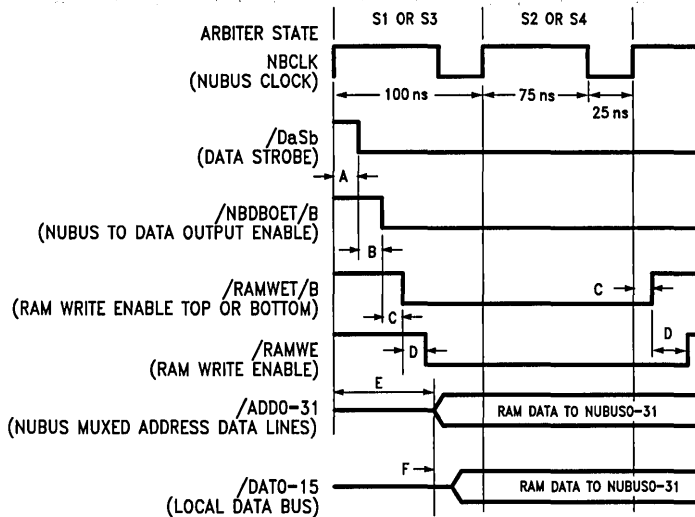


FIGURE A-5. NuBus to RAM Write Timing

TL/F/10805-19



**DATA HOLD AFTER END OF WE**

**Spec Times To Be Met**

Data hold after end of RAMWE = 0 ns min (RAM spec)

**Adapter Card Times**

Data buffer enabling signals are not cleared until after NbClk(S3) goes high

Data hold after end of RAMWE >

$T_{cp}(S2 \text{ or } S4) + T_{cw}(S2 \text{ or } S4) - C - D = 3.5 \text{ ns}$   
(>3.5 ns to spare)

# DP839EB-ATN IBM® PC-AT® Compatible DP83901 SNIC Serial Network Interface Controller Evaluation Board

National Semiconductor  
Application Note 729  
Larry Wakeman



## OVERVIEW

The National Semiconductor SNIC Evaluation Board design provides IBM AT's and AT Compatibles with Thick Ethernet, Thin Ethernet, and Twisted Pair connections. This low parts count evaluation board is compatible with the AT bus and requires only a 1/2 size slot for insertion. Besides using the DP83901 Serial Network Interface Controller (SNIC), the Coaxial Transceiver Interface (CTI) and the Twisted Pair Interface (TPI) are also employed. The dual DMA (local and remote) capabilities of the SNIC, along with 16 kbytes of buffer RAM, allow the entire Network Interface Adapter to appear as a standard I/O Port to the system. The NIC's local DMA channel buffers packets between the local memory (16 kbytes of buffer RAM) and the network, while the NIC's remote DMA channel passes data between the local memory and the system by way of an I/O Port. This I/O Port architecture which isolates the CPU from the network traffic proves to be the simplest method to interface the DP83901 to the system.

## HARDWARE FEATURES

- Fits in half-size IBM PC-AT I/O card form factor
- Utilizes DP83901 Serial Network Interface Controller (SNIC) and DP83922 Twisted Pair Transceiver (TPI)
- 16 kbyte on-board packet buffer

- Simple I/O port interface to IBM PC-AT
- Interfaces to thick Ethernet, thin Ethernet, and twisted pair
- Boot EPROM socket

## NETWORK INTERFACE OPTIONS

The evaluation board supports three physical layer options: Thick Ethernet, Thin Ethernet, and Twisted Pair. These can be seen in *Figure 1*. When using Thick Ethernet, a drop cable is connected to an external transceiver which is in turn connected to a standard Ethernet network. For this physical layer, there is no need for an internal transceiver since it already has an external one. This configuration may be obtained by connecting the pins on JB3 while leaving JB1 and JB2 open. When using Thin Ethernet, a transceiver (the CTI) is available on-board to allow the direct connection to the network via the evaluation board. This transceiver (the CTI) forms the link between the differential ECL signals of the SNI and the non-differential ECL signal of the thin-wire coaxial cable. For proper operation, the CTI requires a DC-DC Converter to provide an isolated ground and a -9V source. In order to put this Thin Ethernet solution into operation, the pins on JB2 need to be connected while JB1 and JB3 should be open.

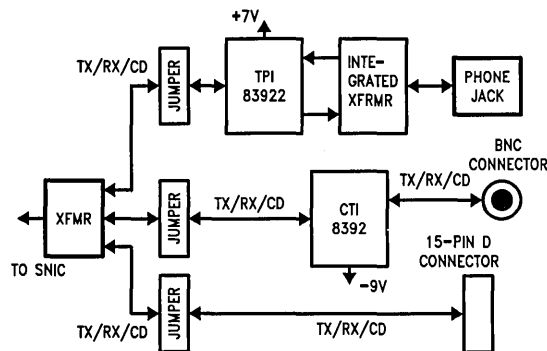


FIGURE 1. Physical Layer Adapter Interface Block Diagram

TL/F/10800-1

When using the Twisted Pair (telephone cable), another transceiver (the TPI) is available on-board which also allows direct connection to the network. The voltage regulator (the LM317) provides the TPI chip with a constant +7V supply. The remaining portion of the TPI circuit includes a common mode choke and a transformer. The transformer decouples the DC component and eliminates any possible voltage spikes. This twisted pair interface can be selected by using JB1 and leaving JB2 and JB3 open. Each of these three physical layer options can be selected simply by the placement of a jumper block (no software changes are needed).

**BUS INTERFACE**

The block diagram, *Figure 2*, illustrates the architecture of the SNIC Evaluation Board. The SNIC Board as seen by the system appears only to be an I/O port. With this architecture the SNIC board has its own local bus to access the board memory. The system never has to intrude further than the I/O ports for any packet data operation. This I/O architecture isolates the system bus and the local bus, thereby preventing interference by the system when the SNIC is doing real-time accesses such as transmitting and receiving packets.

**BOARD ARCHITECTURE**

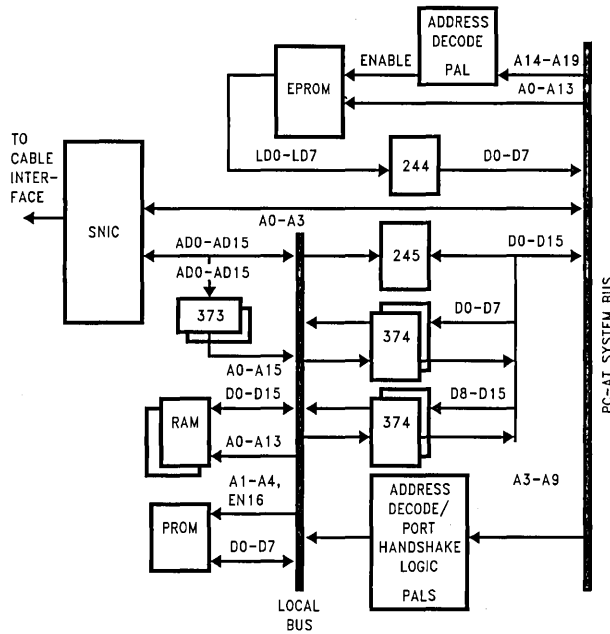
**I/O Map of SNIC Board**

The SNIC Board requires a 32-byte I/O space to allow for decoding the data buffers, the reset port, and the SNIC registers. The first 16 bytes (300h-30Fh) are used to address the SNIC registers (8 bits wide) and the next 8 bytes (310h-317h) are used to address the data buffers which are 16 bits wide. Finally, the reset port may be addressed by 318h-31Fh.

**TABLE I. I/O Map in PC-AT**

Address	Part Addressed
300h-30Fh	SNIC Chip Select
310h-317h	Data Buffers
318h-31Fh	Reset

Although in the description above the I/O map is positioned at the addresses 300-31F, it may also be placed in the following address spaces: 320-33F, 340-35F, 360-37F. These alternate address spaces may be selected by the two jumpers (JP1 and JP0).



**FIGURE 2. Block Diagram of SNIC Evaluation Board's System Interface**

TL/F/10800-2

TABLE II. Optional Address Spaces

Jumpers		
JP1	JP0	I/O Address Space
On	On	300h-31Fh
On	Off	320h-33Fh
Off	On	340h-35Fh
Off	Off	360h-37Fh

**DP83901's Local Memory Map**

There are only two items mapped into the local memory space. These two items being the 8k x 16 buffer RAM and the ID address PROM. The buffer RAM is used for temporary storage of transmit and receive packets. For transmit packets, the remote DMA puts data from the I/O ports into the RAM and the local DMA moves the data from the RAM to the SNIC. For the receive packets, the local DMA carries the data from the SNIC to the RAM and the remote DMA moves the data from the RAM to the I/O ports. The ID address PROM (74S288 32 x 8) contains the physical address of the evaluation board. Each PROM holds its own unique physical address which is installed during its manufacture. The PROM also contains a checksum. This checksum, calculated by exclusive OR-ing the six address bytes with each other, is provided in order to check the addresses. At the initialization of the evaluation board the software commands the SNIC to transfer the PROM data to the I/O Port where it is read by the CPU. The CPU then verifies the checksum and loads the SNIC's physical address registers. The following chart shows the contents of the PROM.

TABLE III. SNIC's Local Memory Map

7FFFh	RAM
4000h	
3FFFh	
0000h	PROM

TABLE IV. PROM Contents

PROM Location	Location Contents
00h	Ethernet Address 0 (Most Significant Byte)
01h	Ethernet Address 1
02h	Ethernet Address 2
03h	Ethernet Address 3
04h	Ethernet Address 4
05h	Ethernet Address 5
06h-0Dh	00h
0Eh	57h
0Fh	57h
10h-15h	Same as 00h-05h
16h-1Dh	00h
1Eh	42h
1Fh	42h

**Data and Address Paths**

For the following paragraph, refer to the block diagram shown in *Figure 2*. Twenty address lines from the PC go onto the SNIC Board, but only four of them actually go to the SNIC. These four addresses along with the NIOR (low-asserted I/O read) or NIOW (low-asserted I/O write) and the CS (SNIC chip select signal) allow the PC to read or write to the SNIC's registers. If the system wants to read from or write to the SNIC registers, the data (only 8 bits) must pass through the 245 buffer. All of the packet data will pass through the I/O ports (the 374's). Each 374 is unidirectional and can only drive 8 bits, therefore it is necessary to have four 374's. Two of which drive data from the ports to the board memory and two of which drive the data from the ports to the AT bus. Even the PROM, which can only be addressed by the SNIC, sends its 8 bits of data out through the 374's. When the PROM does this, two of the 374's will be enabled but only the lower 8 bits will be read by the system. The RAM is also accessed by the SNIC. However, it is addressed by 14 bits and drives out 16 bits of data. The PALs® receive 7 address lines among many other signals such as NIOR, NIOW, NACK, MRD, etc. With these signals the PALs do all of the decodes, such as selecting the SNIC Board, the SNIC chip, the RAM, and the PROM.

**EPROM Socket**

The EPROM socket is provided so that the user may add an EPROM to the board. This EPROM would normally contain a program and a driver to enable the PC-AT to be booted up through the network. The chips necessary to interface the EPROM to the system are the 27128 (EPROM), a 16L8 (PAL), and a 74ALS244 (buffer). Also, JB8 must be placed in the proper selection as described in the jumper section. A PAL decodes SA14-SA19, along with SMRDC (system memory read), in order to generate the EPROMEN signal. This signal, issued when the PC desires to execute the program contained in the EPROM, enables the EPROM and the 244 buffer.

**EVALUATION BOARD OPERATION**

The following pages will describe the slave accesses to the SNIC and the local DMA and remote DMA operation.

**Register Operations**

Accesses to the board are register operations to the DP83901, which are done to set up the SNIC and to control the operation of the SNIC's DMA channels.

**Register Read**

To begin the register read, the CPU drives the four address lines (SA0-SA3) to the SNIC and the SA3-SA9 address lines to the PAL. These address lines are decoded by the PAL in order to generate a chip select to the SNIC. The CPU also drives the NIOR line which the SNIC sees as the NSRD (slave read). Once the SNIC receives this NSRD, it then sends out a high assertion on NACK, acknowledging that it is in slave mode but not yet ready to complete the read. The NACK signal is used by the PAL to assert the IOCHRDY signal false. The SNIC then drives out the data from its internal registers to the 245 buffer. The 245 buffer is then enabled and the data is driven onto the AT BUS. When the SNIC is ready, it asserts NACK true and the PAL asserts IOCHRDY true. As a result, NIOR is driven high by

the CPU, thereby deasserting the NSRD. On the rising edge of the NIOR, the data which is on the AT BUS is latched into the system. The addresses are removed at the same time, causing the SNIC chip select to become deasserted and therefore ending the register read cycle.

#### Register Write

To begin the register write, the CPU drives the SA0-SA3 address lines to the SNIC and the SA4-SA9 address lines to the PAL. With these address lines, the PAL decodes to 300-30F (the SNIC registers) thereby enabling the chip select for the SNIC. The CPU then drives the NIOW strobe which the SNIC sees as NSWR (slave write). Once the SNIC receives this NSWR it sends back a low assertion on NACK to acknowledge that it is in slave mode and ready to perform the write. When the CPU receives this signal, it puts data out onto the AT BUS where it goes into the 245 buffer. The 245 buffer then drives the data to the SNIC, but the data is not latched into the SNIC until the rising edge of NIOW. The system drives NIOW high, thereby deasserting the NSWR and latching the data. The addresses also are taken away and the chip select then goes high (deasserted). This thereby ends the cycle of the register write.

#### Remote Transfers

Remote DMA transfers are operations performed by the SNIC on the board. These operations occur when the SNIC is programmed to transfer packet data between the PC-AT and the card's on-board RAM. These transfers take place through the I/O Port interface.

#### Remote Read

To program the SNIC for a remote read, the CPU must make five slave accesses to the SNIC. The CPU must write the Remote Start Address (2 bytes), the Remote Byte Count (2 bytes) and issue the Remote DMA Read Command. The addresses and byte count require two transfers because they are both 16 bits, yet only 8 bits can be written per transfer.

Once the SNIC has received all of the above data, it drives out BREQ and waits for BACK. For this design the SNIC immediately receives the BACK because it is tied to the BREQ line (BREQ can be tied to BACK because there are no other devices contending for the local bus). After receiving the BACK, the SNIC drives out the address from which the data will be read. This address flows into the 373's and is latched by ADS0. From here, the address flows to the RAM. The RAM waits until it receives NMRD from the SNIC and then it drives the data out of the address it was given and into the 374 ports. The 374 ports then latch the data on the rising edge of the NPWR strobe from the SNIC. PRQ is then sent out by the SNIC to let the system know that there is data waiting in the ports.

If the AT reads the I/O ports before the SNIC has loaded the 374's, then the port request (PRQ) from the SNIC will not yet be driven. This unasserted PRQ signal causes the AT's ready line to be set low, indicating that the SNIC has yet to load the data. After the data is in the ports, the system

must then read the 374 data ports. This begins with the AT driving out an address which is decoded (inside the PAL) to the data I/O Ports (310-317). The PAL then drives RACK to the SNIC, indicating that the CPU is ready to accept data. This RACK signal then reads the data from the 374 ports onto the AT BUS. The system deasserts NIOR which finishes the cycle.

#### Remote Write

Like the remote read, the remote write cycle also begins with five slave accesses into the internal registers. The CPU must write the Remote Start Address (2 bytes), the Remote Byte Count (2 bytes) and issue the Remote DMA Read Command. The SNIC then issues a PRQ. The CPU responds by sending a NIOW, indicating that it is ready to write to the ports. The CPU also drives out the address which corresponds to the I/O Ports. This address goes into the PAL and helps to decode to WACK. This WACK signal latches the data into the 374 ports. The SNIC issues a BREQ and immediately receives a BACK since the two lines are tied together (BREQ can be tied to BACK because there are no other devices contending for the local bus). The SNIC, upon receiving the BACK, drives out address lines to the 373's. These address lines are latched by ADS0 and then are driven to the RAM. SNIC sends out a PRD and a MWR which drives the data from the 374 ports into the already specified address of the onboard memory. Soon afterwards, the PRD and the MWR are deasserted and the cycle ends.

#### Network Transfers

Transfers to and from the network are controlled by the DP83901's local DMA channel which transfers packet data to/from the SNIC's internal FIFO from/to the card's buffer RAM.

#### Receive

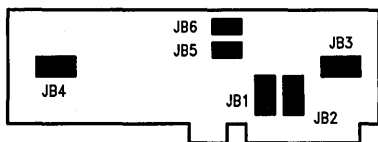
The data comes off of the network, is deserialized and is stored in the FIFO inside of the SNIC. The SNIC then issues a BREQ and immediately receives BACK since the lines are tied together. After receiving BACK, the SNIC drives the address lines to the 373's. The 373's are latched by ADS0 and the address is allowed to flow to the RAM. Then the SNIC drives out NMWR along with the data from the FIFO. The data flows into the RAM at the address given earlier. After this, the NMWR strobe is deasserted thereby causing the cycle to end.

#### Transmit

To begin the transmit cycle, the SNIC issues a BREQ and waits for BACK. Since BREQ and BACK lines are tied together, BACK signal is received immediately. Upon reception of this signal, the SNIC drives out the address to the 373's which latch the address with the ADS0 strobe. The address then flows to the onboard memory. NMRD, driven by SNIC, causes the RAM to drive the data out of the given address and into the SNIC. The SNIC then latches the data into the FIFO on the rising edge of NMRD. This high assertion of NMRD signifies the ending of this cycle. From the FIFO, the data is serialized and transmitted onto the network.

## BOARD CONFIGURATION

On the SNIC-AT board there are six jumper blocks as seen in the diagram below. The following pages will explain how to configure these jumpers.



TL/F/10800-3

### Physical Layer

TABLE V. Physical Layer Selection

JB1	JB2	JB3	Physical Layer Selected
on	off	off	twisted pair
off	on	off	thin ethernet
off	off	on	thick ethernet

If JB1 is closed while JB2 and JB3 are open, then the twisted pair interface will be selected. If JB2 is closed while JB1 and JB3 are open, then the thin ethernet will be selected. And finally if JB3 is closed while JB1 and JB2 are open, then the thick ethernet will be selected.

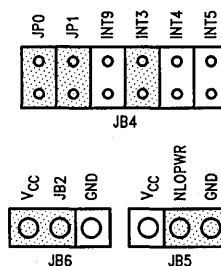
### Interrupt Lines, Board Addresses, and EPROM Addresses

On JB4, there are six possible connections. Four of these are to select an interrupt line. The available interrupt lines include INT3, INT4, INT5, and INT9. The last two possible connections, JP1 and JP0, are used to select the base address for the board. However, if JB6 is connected to V<sub>CC</sub>, then these last two connections select the address of the EPROM also. The possible selections and the jumpers

which should be on (closed) are shown in Table VI. The factory configuration uses the INT3 line for interrupts and has JP1 and JP0 in the on position. This factory configuration is shown in Figure 3, along with the factory configurations for JB5 and JB6.

TABLE VI. Base Address and EPROM Address

JP1	JP0	Base Address	EPROM Address
on	on	300h–31Fh	C800h
on	off	320h–33Fh	CC00h
off	on	340h–35Fh	D000h
off	off	360h–37Fh	D400h



TL/F/10800-4

FIGURE 3. Factory Configuration for JP4, JP5, and JP6

### Low Power SNIC Mode

This low power mode is entirely dependent on JB5. The NLOPWR signal is low-asserted, so in most cases this signal will be jumpered to V<sub>CC</sub>. However, if LAN transmissions are not needed for an extended length of time, the NLOPWR signal may be jumpered to ground. This would turn off the SNI circuitry and conserve power. This feature is primarily provided because the DP83901 enables this function, but can not be practically used in this design.

**PAL EQUATIONS****PAL #1 (U1)**

In this first PAL, the output signals are NIO16, NIOEN, NSNICB, and NCSROM. (The N's before the signals indicate that the signal is low asserted). Since it is necessary to assert NIO16 as soon as possible, this first PAL has been selected to be a 10 ns "D" PAL. The NIO16 signal must be TRI-STATE® when it is not asserted. Therefore, we use an enable signal (NIOEN) which is equal to the decode for the I/O Ports (310-31F) and NAEN high. (NAEN high signifies that the system DMA does not have control of the bus.) The enable signal (NIOEN) loops back into the PAL to bring NIO16 out of TRI-STATE. The NIO16 signal is set to zero so that whenever it is enabled it will be asserted.

The NSNICB signal consists of simple address decodes along with NAEN. The addresses decode to one of four address slots which were mentioned earlier in the board configuration section. The NCSROM is a very simple signal as it consists only of AD14 and NMRD. AD14 comes from the SNIC and selects either the PROM (when low) or the onboard RAM (when high).

PAL 1

```

module iodec;
flag '-rl';
title
date:9/13/89
functions:
SNIC BOARD DECODE, IO16 DECODE, AND CHIP SELECT PROM';
ul device 'p1618';

"input pins:
NEN16, NAEN, SA9      pin 1, 2, 3;
SA8, SA7, SA6        pin 4, 5, 6;
SA5, SA4, SA3        pin 7, 8, 9;
JPO, JP1, NMRD       pin 13, 14, 15;
A14                  pin 16;

"output pins:
NSNICB, NIOEN, NIO16 pin 12, 17, 18;
NCSROM               pin 19;

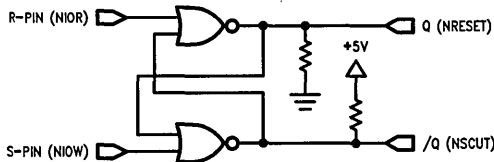
"constants
X = .X.;
Z = .Z.;

equations
NSNICB = !(INAEN & SA9 & SA8 & !SA7 & !SA6 & !SA5 & !JP1 & !JPO
# INAEN & SA9 & SA8 & !SA7 & !SA6 & SA5 & !JP1 & JPO
# !NAEN & SA9 & SA8 & !SA7 & SA6 & !SA5 & JP1 & !JPO
# !NAEN & SA9 & SA8 & !SA7 & SA6 & SA5 & JP1 & JPO);
NIOEN = !(NAEN & SA9 & SA8 & !SA7 & !SA6 & !SA5 & !JP1 & !JPO
& !NEN16 & SA4 & !SA3
# INAEN & SA9 & SA8 & !SA7 & !SA6 & SA5 & !JP1 & JPO
& !NEN16 & SA4 & !SA3
# !NAEN & SA9 & SA8 & !SA7 & SA6 & !SA5 & JP1 & !JPO
& !NEN16 & SA4 & !SA3
# !NAEN & SA9 & SA8 & !SA7 & SA6 & SA5 & JP1 & JPO
& !NEN16 & SA4 & !SA3);
NCSROM = !(A14 & !NMRD);
enable NIO16 = !NIOEN;
NIO16 = 0;
end iodec;

```

## PAL # 2

In this PAL, there are eight outputs which include NRESET, NSOUT, NRDYEN, NIOCHRDY, NCS, NRACK, NWACK and INTO. The first two outputs (NRESET and NSOUT) are part of an R-S flip-flop as shown below:



TL/F/10800-5

FIGURE 4. RS Flip-Flop

NRESET is given by the NOR of the high asserted R-input pin and the NSOUT signal. NSOUT is given by the NOR of the high asserted S-input pin and the NRESET signal. The NOR gates are enabled by the low assertion of NRSTDRV. When the system first boots up, it will disable the NOR gates by asserting the RSTDRV signal. But due to the pull-up and pull-down resistors, the output <NRESET, NSOUT> will be set to <0,1>. Once RSTDRV becomes deasserted, the output will remain at <0,1>. The only way to get out of reset is to assert the S-pin high which is done by an NIOW and an address decode to 318-31F. After the system has booted up, the SNIC may be reset through software. This would be done by setting the R-pin high with an NIOR and an address decode to 318-31F. To escape from reset, we once again set the S-pin high with an NIOW and address decode of 318-31F. The above description of logic is also shown in Truth Table VII.

TABLE VII. R-S Flip-Flop Truth Table

R (NIOR)	S (NIOW)	Q (NRESET)	$\bar{Q}$ (NSOUT)
0	0	0	1
0	1	1	0
0	0	1	0
1	0	0	1
0	0	0	1
0	1	1	0

By using the NIOR and NIOW which are never asserted at the same time, this insures that the R-pin and the S-pin will never be asserted at the same time. The next two signals (NRDYEN and NIOCHRDY) are quite similar to NIOEN and NIO16 in PAL #1. All of the decode takes place in the enable signal (NRDYEN). This decode consists of addresses 300-30F without NACK or the addresses 310-318 without PRQ. If the NRDYEN signal is asserted, then NIOCHRDY will be driven low. At all other times, the NIOCHRDY strobe will be in TRI-STATE. NCS is decoded by NSNICB (from PAL #1) along with the low assertion of SA4 and either NIOR or NIOW. Its decode is in the address range of 300-30F. The last two signals are NRACK and NWACK. NRACK occurs with an address decode to 310-31F, an NIOR, and a PRQ. The NWACK signal only differs from the NRACK by the NIOR/NIOW signal and therefore consists of an address decode to 310-318, an NIOW, and a PRQ. INT is just sent through the PAL to be buffered. The buffered signal which comes out of the PAL is INTO.



```

PAL 2
module reset;
flag '-r1';
title '
date:9/13/89
functions:
RESET LATCH, SNIC SELECT, IOCHRDY, RACK, WACK,
BUFFER INTERRUPT';
u2 device 'p1618';

"input pins:
NSNICB, NIOW, NIOR      pin 1, 2, 3;
RSTDRV, NACK, PRQ      pin 4, 5, 6;
SA4, SA3, INT          pin 7, 8, 9;

"output pins:
INTO, NRACK, NWACK     pin 12, 13, 14;
NRESET, NSOUT, NRDYEN  pin 15, 16, 17;
NIOCHRDY, NCS          pin 18, 19;

"constants
  X = .X.;
  Z = .Z.;

equations
NCS = !(INSNICB & !NIOR & !SA4 # !NSNICB & !NIOW & !SA4);
NRACK = !(INSNICB & PRQ & !NIOR & SA4 & !SA3);
NWACK = !(INSNICB & PRQ & !NIOW & SA4 & !SA3);
NRDYEN = !(INSNICB & !NIOR & !SA4 & NACK
           # !NSNICB & !NIOW & !SA4 & NACK
           # !NSNICB & !PRQ & !NIOR & SA4 & !SA3
           # !NSNICB & !PRQ & !NIOW & SA4 & !SA3);
enable NIOCHRDY = !NRDYEN;
NIOCHRDY = 0;
enable NRESET = !RSTDRV;
NRESET = !(INSNICB & !NIOR & SA4 & SA3 # NSOUT);
enable NSOUT = RSTDRV;
NSOUT = !(NIOW # NRESET);
INTO = INT;
end reset;

```

**PAL #3**

The third PAL only does a decode to enable the optional EPROM. This decode consists of an address decode to C800h, CC00h, D000h, or D400h depending on JP1 and JP0 as shown in the board configuration section. JP2 must also be jumpered for selection of the EPROM. NAEN, a low asserted signal should be high to indicate that the DMA does not have control of the bus and the NSMRDC signal should be asserted low since the CPU is doing a system memory read.

A013 output is used to generate a signal to the lower byte RAM. This signal will route A0 to the RAM when EN16 is high enabling 8-bit operation.

**PAL 3**

```

module epromdec;
  flag '-r0';
  title `
  date:9/13/90
  function:
  EPROM DECODE';

  u21 device 'p1618';

  `input pins:

  A0, A13, EN16                pin 1, 2, 3;
  SMRDC, SA19, SA18            pin 4, 5, 6;
  A17, SA16, SA15              pin 7, 8, 9;
  A14, NAEN, JP2               pin 11, 13, 14;
  P0, JP1                       pin 15, 16;

  `output pins:

  A013                          pin 12;
  EPROMEN                        pin 19;

  `constants
    X = .X.;

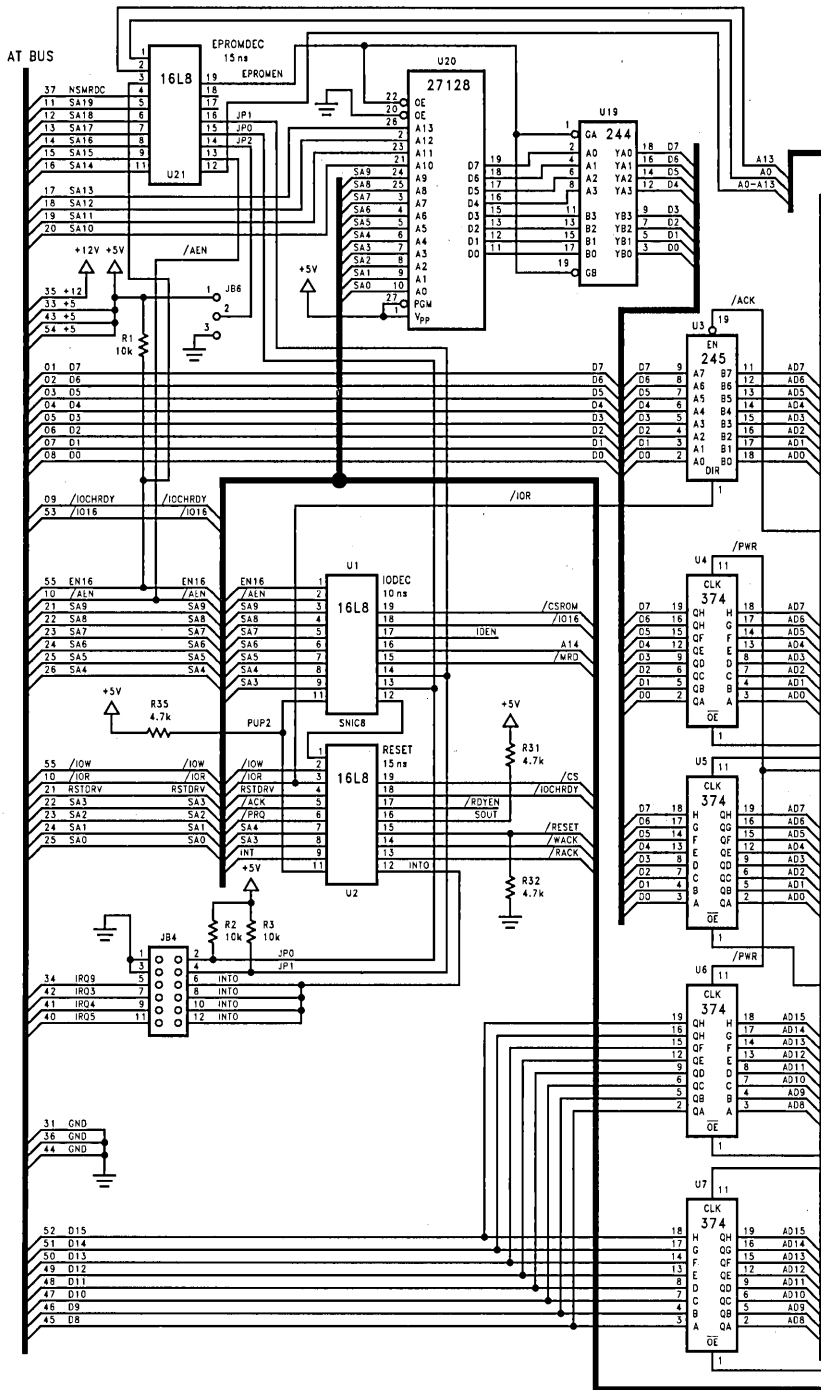
  equations

  NEPROMEN = !(SA19 & SA18 & !SA17 & !SA16 & SA15 & !SA14 & !NAEN
    & JP2 & !JP1 & !JP0 & !NSMRDC
    # SA19 & SA18 & !SA17 & !SA16 & SA15 & SA14 & !NAEN
    & JP2 & !JP1 & JP0 & !NSMRDC
    # SA19 & SA18 & !SA17 & SA16 & !SA15 & !SA14 & !NAEN
    & JP2 & JP1 & !JP0 & !NSMRDC
    # SA19 & SA18 & !SA17 & SA16 & !SA15 & SA14 & !NAEN
    & JP2 & JP1 & JP0 & !NSMRDC);

  A013 = !(A0 & EN16 # A13 & EN16);

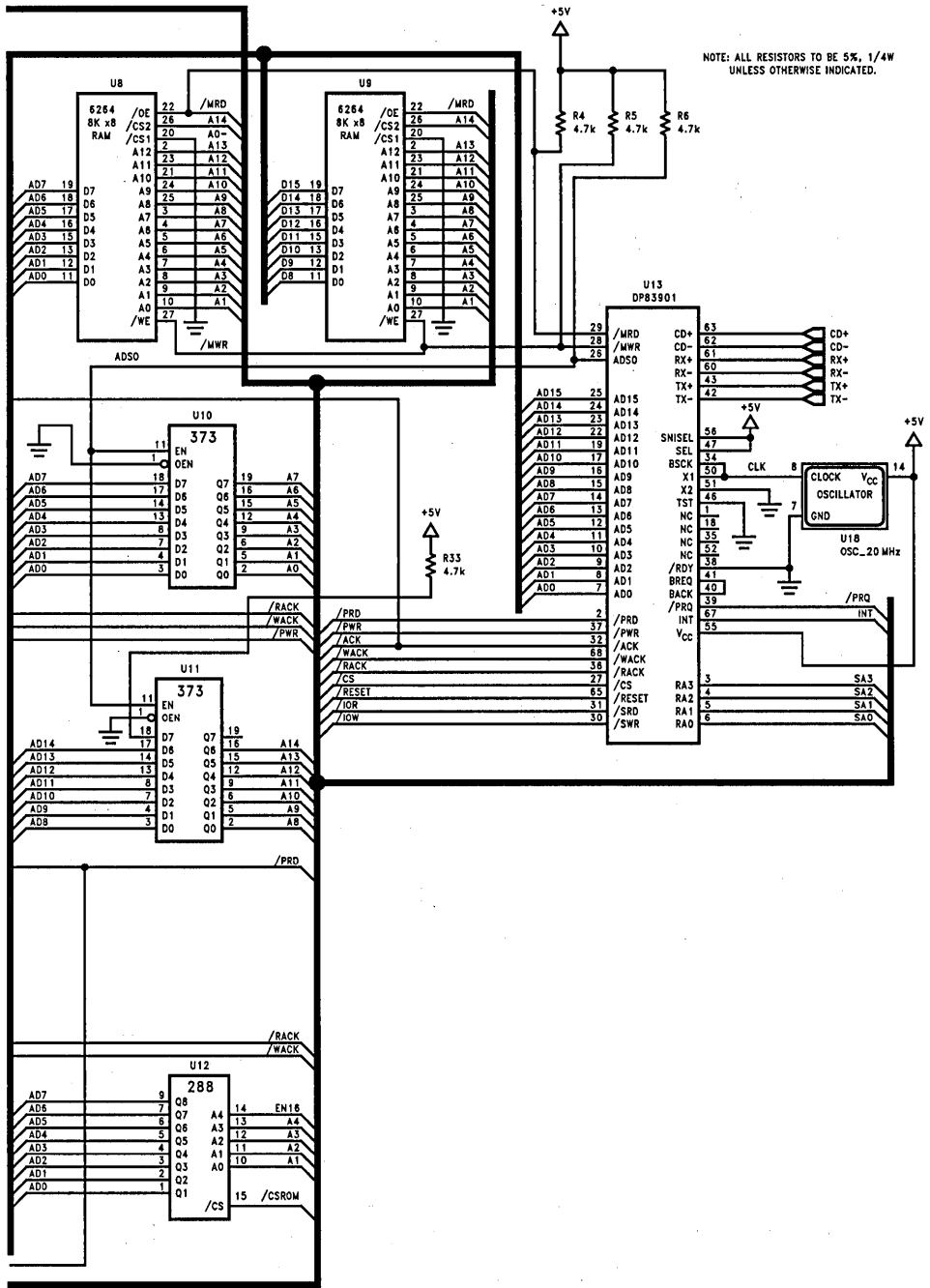
end epromdec;

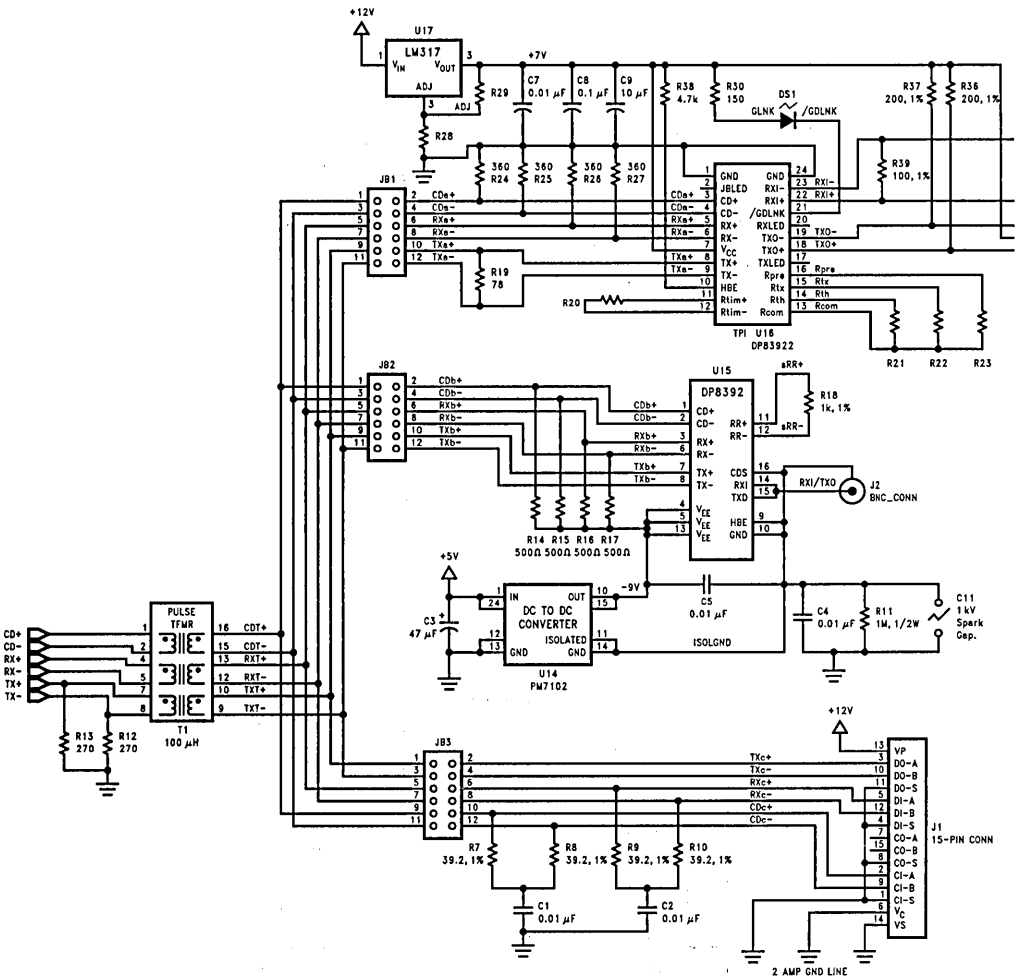
```



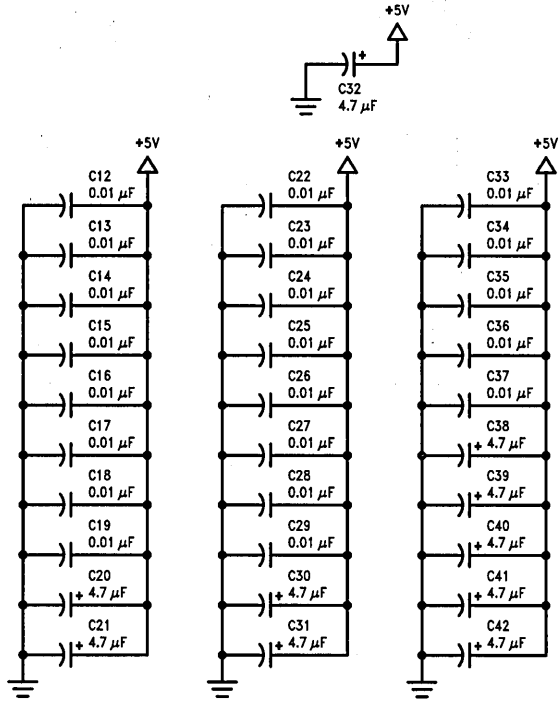
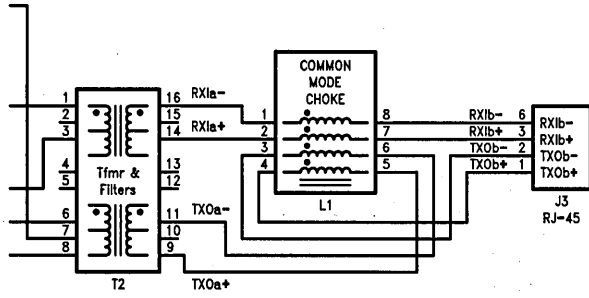
TL/F/10800-6







TL/F/10800-8



TL/F/10800-9

# DP83902EB-AT PC-AT® Compatible DP83902 ST-NIC™ Ethernet Evaluation Board

National Semiconductor  
Application Note 752



AN-752

## OVERVIEW

The National Semiconductor ST-NIC Evaluation Board design provides IBM® AT's and AT Compatibles with Thick Ethernet, Thin Ethernet, and Twisted Pair connections. This low parts count Evaluation Board is compatible with the AT bus and requires only a 1/2 size slot for insertion. The board uses the DP83902 (ST-NIC) to interface to twisted pair Ethernet. The ST-NIC also has an AUI Port which allows interface to thick wire Ethernet, or thin wire Ethernet by the addition of the DP8392 Coaxial Transceiver Interface (CTI). The dual DMA (local and remote) capabilities of the ST-NIC, along with 16 Kbytes of buffer RAM, allow the entire Network Interface Adapter to appear as a standard I/O Port to the system. The NIC module's local DMA channel buffers packets between the local memory (16 Kbytes of buffer RAM) and the network, while the NIC module's remote DMA channel passes data between the local memory and the system by way of an I/O Port. This I/O Port architecture which isolates the CPU from the network traffic proves to be the simplest method to interface the DP83902 to the system.

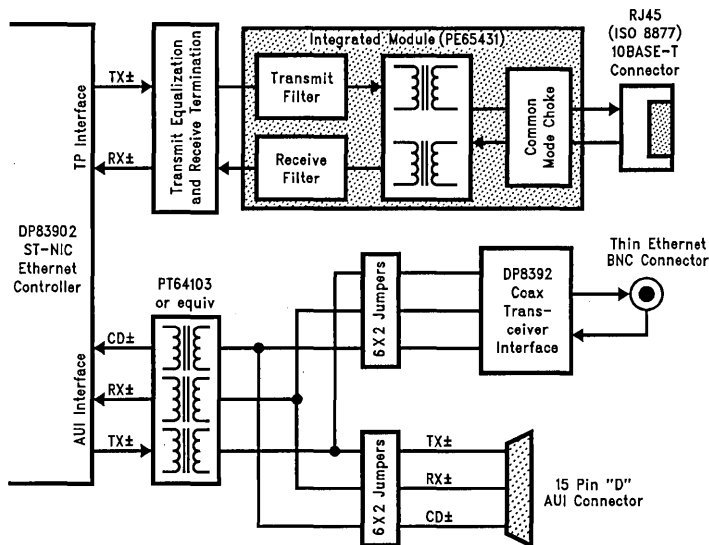
## HARDWARE FEATURES

- Half-size IBM PC-AT I/O Card Form Factor
- Utilizes DP83902 Serial Network Interface Controller for Twisted Pair (ST-NIC)
- 16 Kbyte on-board Packet Buffer
- Simple I/O port interface to IBM PC-AT
- Interfaces to Thick Ethernet, Thin Ethernet, and Twisted Pair
- Boot EPROM Socket

The detailed schematics for this design are shown at the end of this document.

## NETWORK INTERFACE OPTIONS

The evaluation board supports three physical layer options: Thick Ethernet, Thin Ethernet, and Twisted Pair. The block diagram for these interfaces can be seen in *Figure 1*. When using Thick Ethernet, a drop cable is connected to an external transceiver which is in turn connected to a standard Ethernet network, eliminating the need for an internal transceiver. This configuration may be obtained by connecting the pins on JB3 while leaving JB2 open, and connecting JB9 (AUTP) to V<sub>CC</sub>.



TL/F/11158-1

FIGURE 1. Physical Layer Adapter Interface Block Diagram

1

When using Thin Ethernet, a transceiver (the CTI) is available on-board to allow the evaluation board to directly connect to the network. This transceiver (the CTI) forms the link between the differential ECL signals of the SNI module and the non-differential ECL signal of the thin-wire coaxial cable. A DC-DC Converter is provided on the board to supply the CTI with  $-9V$  isolated voltage source. The Thin Ethernet solution is made by connecting the pins on JB2, leaving JB3 open and JB9 (AUTP) should be connected to  $V_{CC}$ .

When using the Twisted Pair, JB9 (AUTP) needs to be connected to ground. The ST-NIC allows direct connection to the network using the RJ-45 phone jack. The remaining circuitry includes pre-emphasis resistors, a filter, a transformer/filter and a common mode choke. The transformer/filter decouples the DC component and eliminates any possible voltage spikes.

The diagram in *Figure 2* illustrates the layout of the board. It shows the various jumpers, ICs, LEDs, and the connectors for the three physical layer options. The transmit pre-emphasis resistors R27-R30 provide equalization to the twisted pair transmit outputs. This boosts the higher harmonics of the signal in order to compensate for losses in these harmonics over the twisted pair cable. R19 and R20 are  $50\Omega$  each and when combined form the required  $100\Omega$  termination on the receive side.

#### BUS INTERFACE

The block diagram, *Figure 3*, illustrates the architecture of the ST-NIC Evaluation Board. The ST-NIC Board as seen by the system appears only to be an I/O port. With this architecture the ST-NIC board has its own local bus to access the board memory. The system never has to intrude further than the I/O ports for any packet data operation. This I/O architecture isolates the system bus and the local bus, thereby preventing interference by the system when the ST-NIC is doing real-time accesses such as transmitting and receiving packets.

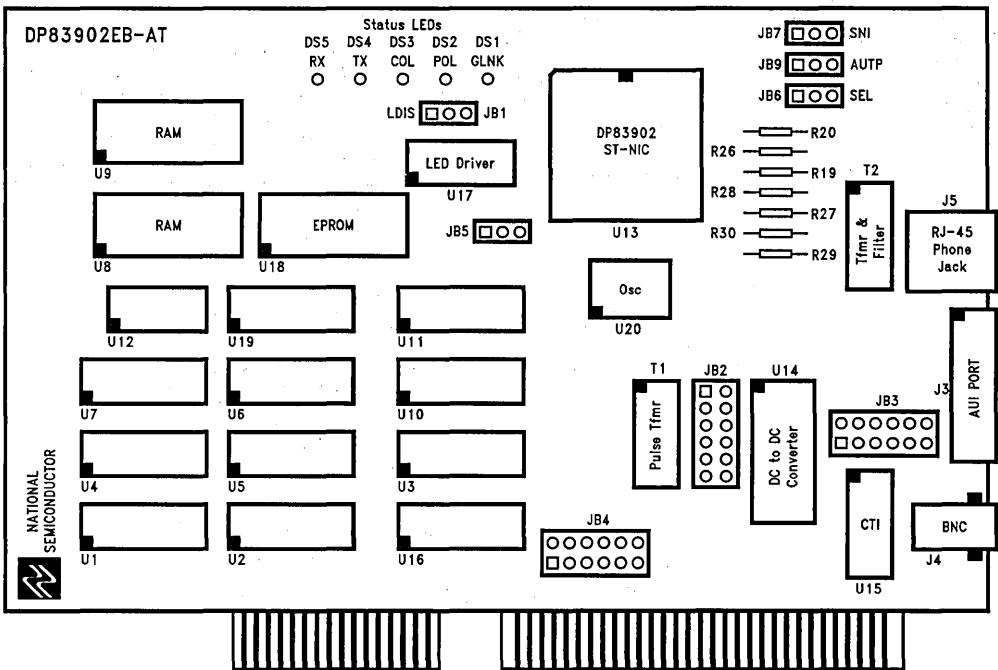


FIGURE 2. Layout of ST-NIC Evaluation Board

TL/F/11158-2



**BOARD ARCHITECTURE**

**I/O Map of ST-NIC Board**

The ST-NIC Board requires a 32-byte I/O space to allow for decoding the data buffers, the reset port, and the ST-NIC registers. The first 16 bytes (300h-30Fh) are used to address the ST-NIC registers (8 bits wide) and the next 8 bytes (310h-317h) are used to address the data buffers which are 16 bits wide. Finally, the reset port (also software selectable) may be addressed by 318h-31Fh.

**TABLE I. I/O MAP in PC-AT**

Address	Part Addressed
300h-30Fh	ST-NIC Chip Select
310h-317h	Data Buffers
318h-31Fh	Reset

Although in the description above the I/O map is positioned at the addresses 300-31F, it may also be placed in the following address spaces: 320-33F, 340-35F, 360-37F.

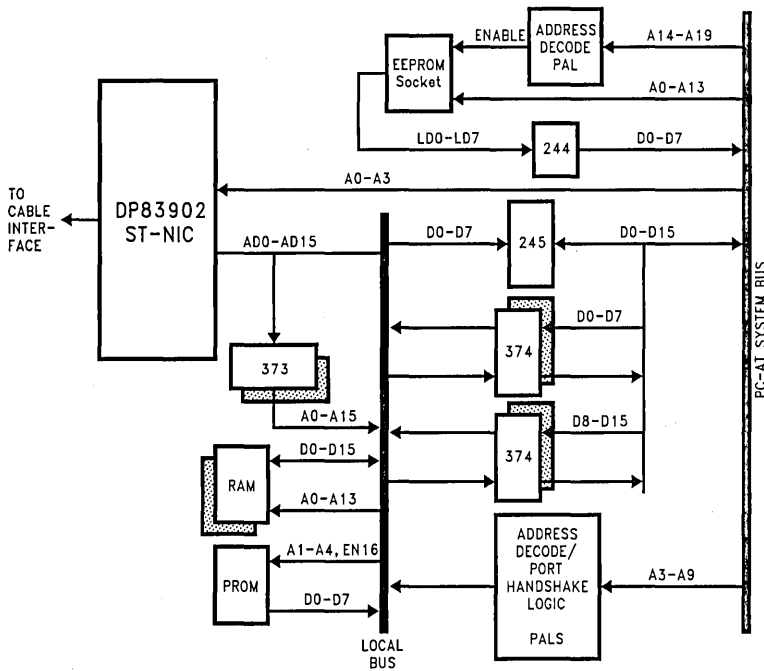
These alternate address spaces may be selected by the two jumper pins JP1 and JP0 (refer to JB4 in *Figure 4* and Appendix A).

**DP83902's Local Memory Map**

There are only two items mapped into the local memory space. These two items being the 8K x 16 buffer RAM and the ID address PROM. The buffer RAM is used for temporary storage of transmit and receive packets.

**TABLE II. ST-NIC's Local Memory Map**

7FFFh	RAM
4000h	
3FFFh	PROM
0000h	



**FIGURE 3. Block Diagram of ST-NIC Evaluation Board's System Interface**

TL/F/111158-3

For transmit packets, the remote DMA puts data from the I/O ports into the RAM and the local DMA moves the data from the RAM to the ST-NIC. For the receive packets, the local DMA carries the data from the ST-NIC to the RAM and the remote DMA moves the data from the RAM to the I/O ports. The ID address PROM (74S288 32 x 8) contains the physical address of the evaluation board. Each PROM holds its own unique physical address which is installed during its manufacture. The PROM also contains some identification bytes that can be checked by the driver software. At the initialization of the evaluation board the software commands the ST-NIC to transfer the PROM data to the I/O Port where it is read by the CPU. The CPU then loads the ST-NIC's physical address registers. The following chart shows the contents of the PROM.

**TABLE III. PROM Contents**

PROM Location	Location Contents
00h	Ethernet Address 0 (Most Significant Byte)
01h	Ethernet Address 1
02h	Ethernet Address 2
03h	Ethernet Address 3
04h	Ethernet Address 4
05h	Ethernet Address 5
06h-0Dh	00h
0Eh, 0Fh	57h
10h-15h	Ethernet Address 0 thru 5
16h-1Dh	Reserved
1Eh, 1Fh	42h

#### Data and Address Paths

For the following paragraph, refer to the block diagram shown in *Figure 3*. Twenty address lines from the PC go onto the ST-NIC Board, but only four of them actually go to the ST-NIC. These four addresses along with the NIOR (low-asserted I/O read) or NIOW (low-asserted I/O write) and the CS (ST-NIC chip select signal) allow the PC to read or write to the ST-NIC's registers. If the system wants to read from or write to the ST-NIC registers, the data (only 8 bits) must pass through the 245 buffer. All of the packet data will pass through the I/O ports (the 374's). Each 374 is unidirectional and can only drive 8 bits, therefore it is necessary to have four 374's. Two of which drive data from the ports to the board memory and two of which drive the data from the ports to the AT bus. Even the PROM, which can only be addressed by the ST-NIC, sends its 8 bits of data out through the 374's. When the PROM does this, two of the 374's will be enabled but only the lower 8 bits will be read by the system. The RAM is also accessed by the ST-NIC. However, it is addressed by 14 bits and drives out 16 bits of data. The PALs receive 7 address lines among many other signals such as NIOR, NIOW, NACK, MRD, etc. With these signals the PALs do all of the decodes, such as selecting the ST-NIC Board, the ST-NIC chip, the RAM, and the PROM.

#### EPROM SOCKET

The EPROM socket is provided so that the user may add an EPROM to the system. This EPROM would normally contain a program and a driver to enable the PC-AT to be booted up through the network. The chips necessary to interface the EPROM to the system are the 27128 (EPROM), a 16L8 (PAL), and a 74ALS244 (buffer). Also, JB5 must be placed in the proper selection as described in the jumper section. The PAL decodes SA14-SA19, along with SMRDC (system memory read), in order to generate the EPROMEN signal. This signal, issued when the PC wants to execute the program stored in the EPROM, enables the EPROM and the 244 buffer.

#### EVALUATION BOARD OPERATION

The following pages will describe the slave accesses to the ST-NIC and the local DMA and remote DMA operation.

#### Register Operations

Accesses to the board are register operations to the DP83902, which are done to set up the ST-NIC and to control the operation of the ST-NIC's DMA channels.

#### REGISTER READ

To begin the register read, the CPU drives the four address lines (SA0-SA3) to the ST-NIC and the SA3-SA9 address lines to the PAL. These address lines are decoded by the PAL in order to generate a chip select to the ST-NIC. The CPU also drives the NIOR line which the ST-NIC sees as the NSRD (slave read). Once the ST-NIC receives this NSRD, it then sends out a high assertion on NACK, acknowledging that it is in slave mode but not yet ready to complete the read. The NACK signal is used by the PAL to assert the IOCHRDY (used to insert wait states) signal false. The ST-NIC then drives out the data from its internal registers to the 245 buffer. The 245 buffer is then enabled and the data is driven onto the AT BUS. When the ST-NIC is ready, it asserts NACK true and the PAL asserts IOCHRDY true. As a result, NIOR is driven high by the CPU, thereby deasserting the NSRD. On the rising edge of the NIOR, the data which is on the AT BUS is latched into the system. The addresses are removed at the same time, causing the ST-NIC chip select to become deasserted, ending the register read cycle.

#### REGISTER WRITE

To begin the register write, the CPU drives the SA0-SA3 address lines to the ST-NIC and the SA4-SA9 address lines to the PAL. With these address lines, the PAL decodes to 300-30F (the ST-NIC registers) thereby enabling the chip select for the ST-NIC. The CPU then drives the NIOW strobe which the ST-NIC sees as NSWR (slave write). Once the ST-NIC receives this NSWR it sends back a low assertion on NACK to acknowledge that it is in slave mode and ready to perform the write. When the CPU receives this signal, it puts data out onto the AT BUS where it goes into the 245 buffer. The 245 buffer then drives the data to the ST-NIC, but the data is not latched into the ST-NIC until the rising edge of NIOW. The system drives NIOW high, thereby deasserting the NSWR and latching the data. The addresses also are taken away and the chip select then goes high (deasserted). This ends the cycle of the register write.

## Remote Transfers

Remote DMA transfers are operations performed by the ST-NIC on the board. These operations occur when the ST-NIC is programmed to transfer packet data between the PC-AT and the card's on-board RAM. These transfers take place through the I/O Port interface.

### REMOTE READ

To program the ST-NIC for a remote read, the CPU must make five slave accesses to the ST-NIC. The CPU must write the Remote Start Address (2 bytes), the Remote Byte Count (2 bytes), and issue the Remote DMA Read Command. The addresses and byte count require two transfers because they are both 16 bits, yet only 8 bits can be written per transfer.

Once the ST-NIC has received all of the above data, it drives out BREQ and waits for BACK. The ST-NIC immediately receives BACK because it is tied to the BREQ line. (BREQ can be tied to BACK because there are no other devices contending for the local bus.) After receiving BACK, the ST-NIC drives out the address from which the data is required to be read. This address flows into the 373's and is latched by ADS0. From here, the address flows to the RAM. The RAM waits until it receives MRD from the ST-NIC and then it drives the data into the 374 ports. The 374 ports then latch the data on the rising edge of the PWR strobe from the ST-NIC. PRQ is then sent out by the ST-NIC to let the system know that there is data waiting in the ports.

If the AT reads the I/O ports before the ST-NIC has loaded the 374's, then the port request (PRQ) from the ST-NIC will not yet be driven. This unasserted PRQ signal causes the AT's ready line to be set low, indicating that the ST-NIC has yet to load the data. After the data is in the ports, the system must then read the 374 data ports. This begins with the AT driving out an address which is decoded (inside the PAL) to the data I/O Ports (310–31F). The PAL then drives RACK to the ST-NIC, indicating that the CPU is ready to accept data. This RACK signal then reads the data from the 374 ports onto the AT BUS. The system deasserts NIOR which finishes the cycle.

### REMOTE WRITE

Like the remote read, the remote write cycle also begins with five slave accesses into the internal registers. The CPU must write the Remote Start Address (2 bytes), the Remote Byte Count (2 bytes), and issue the Remote DMA Write Command. The ST-NIC then issues a PRQ. The CPU responds by sending an NIOW, indicating that it is ready to write to the ports. The CPU also drives out the address which corresponds to the I/O Ports. This address goes into the PAL and helps to decode to WACK. This WACK signal latches the data into the 374 ports. The ST-NIC issues a BREQ and immediately receives a BACK since the two lines are tied together. (BREQ can be tied to BACK because there are no other devices contending for the local bus.) The ST-NIC, upon receiving the BACK, drives out address lines to the 373's. These address lines are latched by ADS0 and then are driven to the RAM. ST-NIC sends out a PRD and a NMWR which drives the data from the 374 ports into the already specified address of the onboard memory. PRD and NMWR are then deasserted and the cycle ends.

## Network Transfers

Transfers to and from the network are controlled by the DP83902's local DMA channel which transfers packet data to/from the ST-NIC's internal FIFO from/to the card buffer's RAM.

### RECEIVE

The data comes off of the network, is deserialized and is stored in the FIFO inside of the ST-NIC. The ST-NIC then issues a BREQ and immediately receives BACK since the lines are tied together. After receiving BACK, the ST-NIC drives the address lines to the 373's. The 373's are latched by ADS0 and the address is allowed to flow to the RAM. Then the ST-NIC drives out NMWR along with the data from the FIFO. The data flows into the RAM under the address given earlier. The NMWR strobe is then deasserted, ending the cycle.

### TRANSMIT

To begin the transmit cycle, the ST-NIC issues a BREQ and waits for the BACK. Since the BREQ and BACK lines are tied together, the BACK signal is received immediately. Upon reception of this signal, the ST-NIC drives out the address to the 373's which latch the address with the ADS0 strobe. The address then flows to the onboard memory. NMRD, driven by ST-NIC, causes the RAM to drive the data out of the given address and into the ST-NIC. The ST-NIC then latches the data into the FIFO on the rising edge of NMRD. This high assertion of NMRD signifies the ending of this cycle. From the FIFO, the data is serialized and transmitted onto the network.

## BOARD CONFIGURATION

On the DP83902EB-AT ST-NIC AT board, there are nine jumper blocks as seen in the diagram below. The following pages will explain how to configure these jumpers.

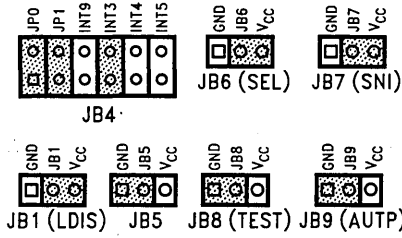
### Physical Layer

If JB9 is tied to Ground then the twisted pair interface will be selected. If JB2 is closed while JB3 is open, and JB9 is connected to V<sub>CC</sub>, then the Thin Ethernet option will be selected. And finally if JB3 is closed while JB2 is open, and JB9 is high, then the Thick Ethernet option will be selected. Refer to Appendix A for Jumper settings.

### Interrupt Lines, Board Addresses, and EPROM Addresses

On JB4, there are six possible connections. Four of these are to select an interrupt line. The available interrupt lines include INT3, INT4, INT5, and INT9. The last two possible connections, JP1 and JP0, are used to select the base address for the board. However, if JB5 is connected to V<sub>CC</sub>, then these last two connections select the address of the EPROM also. The possible selections and the jumpers are shown in Appendix A. The factory configuration uses the INT3 line for interrupts and has JP1 and JP0 in the on position.

This factory configuration is shown in *Figure 4*, along with the factory configurations for JB1, JB5, JB6, JB7, JB8, and JB9. The square pin indicates pin 1 of the jumpers.



TL/F/11158-4

**FIGURE 4. Factory Configuration for JB1, JB4, JB5, JB6, JB7, JB8 and JB9**

**APPENDIX A**

The following tables show all of the various jumper settings. The shaded boxes are the Factory Configuration default settings.

<b>JB1</b>	High	Link Enabled
	Low	Link Disabled
<b>JB5</b>	High	EPROM Address
	Low	Base Address
<b>JB6</b>	High	Tx+ and Tx- are same in idle state
	Low	Tx+ is positive with respect to Tx- in idle state
<b>JB7</b>	High	Normal Operation
	Low	ENDEC Module Testing
<b>JB8</b>	High	Internal Function Testing
	Low	Normal Operation

<b>JB9</b>	<b>JB2</b>	<b>JB3</b>	<b>Physical Layer Selected</b>
Low	X	X	Twisted Pair
High	On	Off	Thin Ethernet
High	Off	On	Thick Ethernet

<b>INT9</b>	<b>INT3</b>	<b>INT4</b>	<b>INT5</b>	<b>Interrupt Selection</b>
On	Off	Off	Off	Interrupt 9
Off	On	Off	Off	Interrupt 3
Off	Off	On	Off	Interrupt 4
Off	Off	Off	On	Interrupt 5

<b>JP1</b>	<b>JP0</b>	<b>Base Address</b>	<b>EPROM Address</b>
On	On	300h-31Fh	C800h
On	Off	320h-33Fh	CC00h
Off	On	340h-35Fh	D000h
Off	Off	300h-37Fh	D400h

## APPENDIX B: PAL® EQUATIONS

## PAL #1 (U1)

In this first PAL, the output signals are NIO16, NIOEN, NSTNICB, and NCSROM. (The N's before the signals indicate that the signal is low asserted.) Since it is necessary to assert NIO16 as soon as possible, this first PAL has been selected to be a 10 ns "D" PAL. The NIO16 signal must be TRI-STATE® when it is not asserted. Therefore, we use an enable signal (NIOEN) which is equal to the decode for the I/O Ports (310-31F) and NAEN high (NAEN high signifies that the system DMA does not have control of the bus). The

enable signal (NIOEN) loops back into the PAL to bring NIO16 out of TRI-STATE. The NIO16 signal is set to zero so that whenever it is enabled it will be asserted.

The STNICB signal consists of simple address decodes along with NAEN. The addresses decode to one of four address slots which were mentioned earlier in the board configuration section. The NCSROM is a very simple signal as it consists only of AD14 and NMRD. AD14 comes from the ST-NIC and selects either the PROM (when low) or the onboard RAM (when high).

## PAL 1

```

module iodec;
flag '-r1';
title `
date: 9/13/89
functions:
ST-NIC BOARD DECODE, IO16 DECODE, AND CHIP SELECT PROM';
ul device 'p1618';

`input pins:
NEN16, NAEN, SA9      pin 1, 2, 3;
SA8, SA7, SA6        pin 4, 5, 6;
SA5, SA4, SA3        pin 7, 8, 9;
JP0, JP1, NMRD       pin 13, 14, 15;
A14                  pin 16;

`output pins:
NSTNICB, NIOEN, NIO16 pin 12, 17, 18;
NCSROM               pin 19;

`constants
X = .X.;
Z = .Z.;

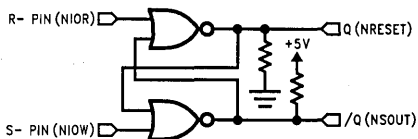
equations
NSTNICB = (!(NAEN & SA9 & SA8 & !SA7 & !SA6 & !SA5 & !JP1 & !JP0
# !NAEN & SA9 & SA8 & !SA7 & !SA6 & SA5 & !JP1 & JP0
# !NAEN & SA9 & SA8 & !SA7 & SA6 & !SA5 & JP1 & !JP0
# !NAEN & SA9 & SA8 & !SA7 & SA6 & SA5 & JP1 & JP0);
NIOEN = (!(NAEN & SA9 & SA8 & !SA7 & !SA6 & !SA5 & !JP1 & !JP0
& !NEN16 & SA4 & !SA3
# !NAEN & SA9 & SA8 & !SA7 & !SA6 & SA5 & !JP1 & JP0
& !NEN16 & SA4 & !SA3
# !NAEN & SA9 & SA8 & !SA7 & SA6 & !SA5 & JP1 & !JP0
& !NEN16 & SA4 & !SA3
# !NAEN & SA9 & SA8 & !SA7 & SA6 & SA5 & JP1 & JP0
& !NEN16 & SA4 & !SA3);
NCSROM = (!(A14 & !NMRD);
enable NIO16 = !NIOEN;
NIO16 = 0;
end iodec;

```

TL/F/11158-5

**PAL #2**

In this PAL there are eight outputs: NRESET, NSOUT, NRDYEN, NIOCHRDY, NCS, NRACK, NWACK and INTO. The first two outputs (NRESET and NSOUT) are part of an R-S flip flop as shown below:



TL/F/11158-6

**FIGURE 5. RS Flip-Flop**

NRESET is given by the NOR of the high asserted R-input pin and the NSOUT signal. NSOUT is given by the NOR of the high asserted S-input pin and the NRESET signal. The NOR gates are enabled by the low assertion of NRSTDRV. When the system first boots up, it will disable the NOR gates by asserting the RSTDRV signal. But due to the pull-up and pull-down resistors, the output <NRESET, NSOUT> will be set to <0, 1>. Once RSTDRV becomes deasserted, the output will remain at <0, 1>. The only way to get out of reset is to assert the S-pin high which is done by an NIOW and an address decode to 318-31F. After the system has booted up, the ST-NIC may be reset through software. This would be done by setting the R-pin high with an NIOR and an address decode to 318-31F. To escape from reset, we once again set the S-pin high with an NIOW and address decode of 318-31F. The above description of logic is also shown in Truth Table VII.

**TABLE IV. R-S Flip Flop Truth Table**

R (NIOR)	S (NIOW)	Q (NRESET)	$\bar{Q}$ (NSOUT)
0	0	0	1
0	1	1	0
0	0	1	0
1	0	0	1
0	0	0	1
0	1	1	0

By using the NIOR and NIOW which are never asserted at the same time, this insures that the R-pin and the S-pin will never be asserted at the same time. The next two signals (NRDYEN and NIOCHRDY) are quite similar to NIOEN and NIO16 in PAL#1. All of the decode takes place in the enable signal (NRDYEN). This decode consists of addresses 300-30F without NACK or the addresses 310-318 without PRQ. If the NRDYEN signal is asserted, then NIOCHRDY will be driven low. At all other times, the NIOCHRDY strobe will be in TRI-STATE. This PAL must also be a 10 ns "D" PAL.

NCS is decoded by NSTNICB (from PAL #1) along with the low assertion of SA4 and either NIOR or NIOW. Its decode is in the address range of 300-30F. The last two signals are NRACK and NWACK. NRACK occurs with an address decode to 310-318, an NIOR, and a PRQ. The NWACK signal only differs from the NRACK by the NIOR/NIOW signal and therefore consists of an address decode to 310-31F, an NIOW and a PRQ. INT is just sent through the PAL to be buffered. The buffered signal which comes out of the PAL is INTO.

PAL2

```

module reset;
flag '-rl';
title `
date: 9/13/89
functions:
RESET LATCH, STNIC SELECT, IOCHRDY, RACK, WACK,
BUFFER INTERRUPT';
u2 device 'p1618';

```

```

`input pins:
NSTNICB, NIOW, NIOR      pin 1, 2, 3;
RSTDRV, NACK, PRQ       pin 4, 5, 6;
SA4, SA3, INT           pin 7, 8, 9;

```

```

`output pins:
INTO, NRACK, NWACK      pin 12, 13, 14;
NRESET, NSOUT, NRDYEN   pin 15, 16, 17;
NIOCHRDY, NCS           pin 18, 19;

```

```

`constants
X = .X.;
Z = .Z.;

```

```

equations
NCS = !(NSTNICB & !NIOR & !SA4 # !NSTNICB & !NIOW & !SA4);
NRACK = !(NSTNICB & PRQ & !NIOR & SA4 & !SA3);
NWACK = !(NSTNICB & PRQ & !NIOW & SA4 & !SA3);
NRDYEN = !(NSTNICB & !NIOR & !SA4 & NACK
# !NSTNICB & !NIOW & !SA4 & NACK
# !NSTNICB & !PRQ & !NIOR & SA4 & !SA3
# !NSTNICB & !PRQ & !NIOW & SA4 & !SA3);
enable NIOCHRDY = !NRDYEN;
NIOCHRDY = 0;
enable NRESET = !RSTDRV;
NRESET = !(NSTNICB & !NIOR & SA4 & SA3 # NSOUT);
enable NSOUT = !RSTDRV;
NSOUT = !(NIOW # NRESET);
INTO = INT;
end reset;

```

TL/F/11158-7

**PAL #3**

The third PAL only does a decode to enable the optional EPROM. This decode consists of an address decode to C800h, CC00h, D000h, or D400h depending on JP1 and JP0 as shown in the board configuration section. JP2 must

also be jumpered for selection of the EPROM. NAEN, a low asserted signal should be high to indicate that the DMA does not have control of the bus and the NSMRDC signal should be asserted high since the CPU is doing a system memory read.

**PAL 3**

```

module epromdec;
flag '-r0';
title `
date: 9/13/89
function:
EPROM DECODE';

u16 device 'p1618';

`input pins:

A0, A13, EN16           pin 1, 2, 3;
SMRDC, SA19, SA18      pin 4, 5, 6;
A17, SA16, SA15        pin 7, 8, 9;
A14, NAEN, JP2         pin 11, 13, 14;
P0, JP1                pin 15, 16;

`output pins:

EPROMEN                 pin 19;
A013                    pin 12;

`constants
X = .X.;

equations

NEPROMEN = !(SA19 & SA18 & !SA17 & !SA16 & SA15 & !SA14 & !NAEN
& JP2 & !JP1 & !JP0 & !NSMRDC
# SA19 & SA18 & !SA17 & !SA16 & SA15 & SA14 & !NAEN
& JP2 & !JP1 & JP0 & !NSMRDC
# SA19 & SA18 & !SA17 & SA16 & !SA15 & !SA14 & !NAEN
& JP2 & JP1 & !JP0 & !NSMRDC
# SA19 & SA18 & !SA17 & SA16 & !SA15 & SA14 & !NAEN
& JP2 & JP1 & JP0 & !NSMRDC);

A013 = !(A0 & EN16 # A13 & !EN16);

end epromdec;

```

TL/F/11158-8



**APPENDIX C: BILL OF MATERIALS FOR DP83902EB-AT ST-NIC ETHERNET ADAPTER****CAPACITORS**

C1, C20..01	μF	50V	10%	Monolythic
C3	4.7	μF	25V	20% Tantalum
C4	0.01	μF	1 KV	10% Ceramic Disk
C5	0.01	μF	50V	10% Ceramic Disk
C9..C10	0.01	μF	50V	20% Monolythic
C11	0.75	pF	1 KV	Spark Gap
C12..C19	0.01	μF	50V	20% Monolythic
C20	4.7	μF	25V	20% Tantalum
C21	4.7	μF	25V	20% Tantalum
C22..C29	0.01	μF	50V	20% Monolythic
C30..C32	4.7	μF	25V	20% Tantalum
C33..C37	0.01	μF	50V	20% Monolythic
C38..C42	4.7	μF	25V	20% Tantalum

**RESISTORS**

R1..R3	10K		1/4W	5%
R4..R6	4.7K		1/4W	5%
R7..R10	39.2		1/4W	1%
R11	1M		1/2W	5%
R12, R13	270		1/4W	5%
R14..R17	1.5K		1/4W	5%
R18	1K		1/4W	1%
R19..R20	TBD		1/4W	1%
R21	420		1/4W	5%
R22..R25	430		1/4W	5%
R26	4.7K		1/4W,	5%
R27..R30	TBD		1/4W	1%
R31..R34	4.7K		1/4W	5%

**IC's**

U1, U2	16L8D		PAL
U16	16L8B		PAL
U3	74ALS245		
U4..U7	74ALS374		
U8, U9	HM6264		8K x 8 STATIC RAM 100 ns
U10, U11	74AS373		
U12	74S288		PROM
U13	DP83902		ST-NIC
U15	DP8392		CTI
U17	74HC04N		
U18	27128		EPR0M (not supplied on board)
U19	74ALS244		
U20	20 MHz 0.01%		Crystal Oscillator

**Note:****ETHERNET ID PROM ADDRESS ASSIGNMENT:**

Registration Authority for ISO/IEC 8802-3  
 c/o The Institute of Electrical and Electronics Engineers  
 445 Hoes Lane  
 P.O. Box 1331  
 Piscataway, NJ 08055-1331  
 (908) 562-3812

**MAGNETICS (TRANSFORMER, FILTER, CHOKE, DC-DC CONVERTOR, ETC.)**

See Section 5 of databook, Ethernet Magnetics Vendors

**SPARK GAP SUPPLIERS:**

0.75 pFkV Spark Gap

**Mallory Part# ASR75A**  
 (317) 856-3731

**Mepco/Centralab Part# S758X44000NAZAA**  
 Available from: Philips Components Discrete Product Division  
 (602) 820-2225

**MAGNETICS**

U14 PM7102 VALOR DC-DC Converter.  
 T1 PE64103 Pulse Engineering  
 T2 RX and TX Filter & Transformer. Pulse Engineering PE65431.

**MISCELLANEOUS**

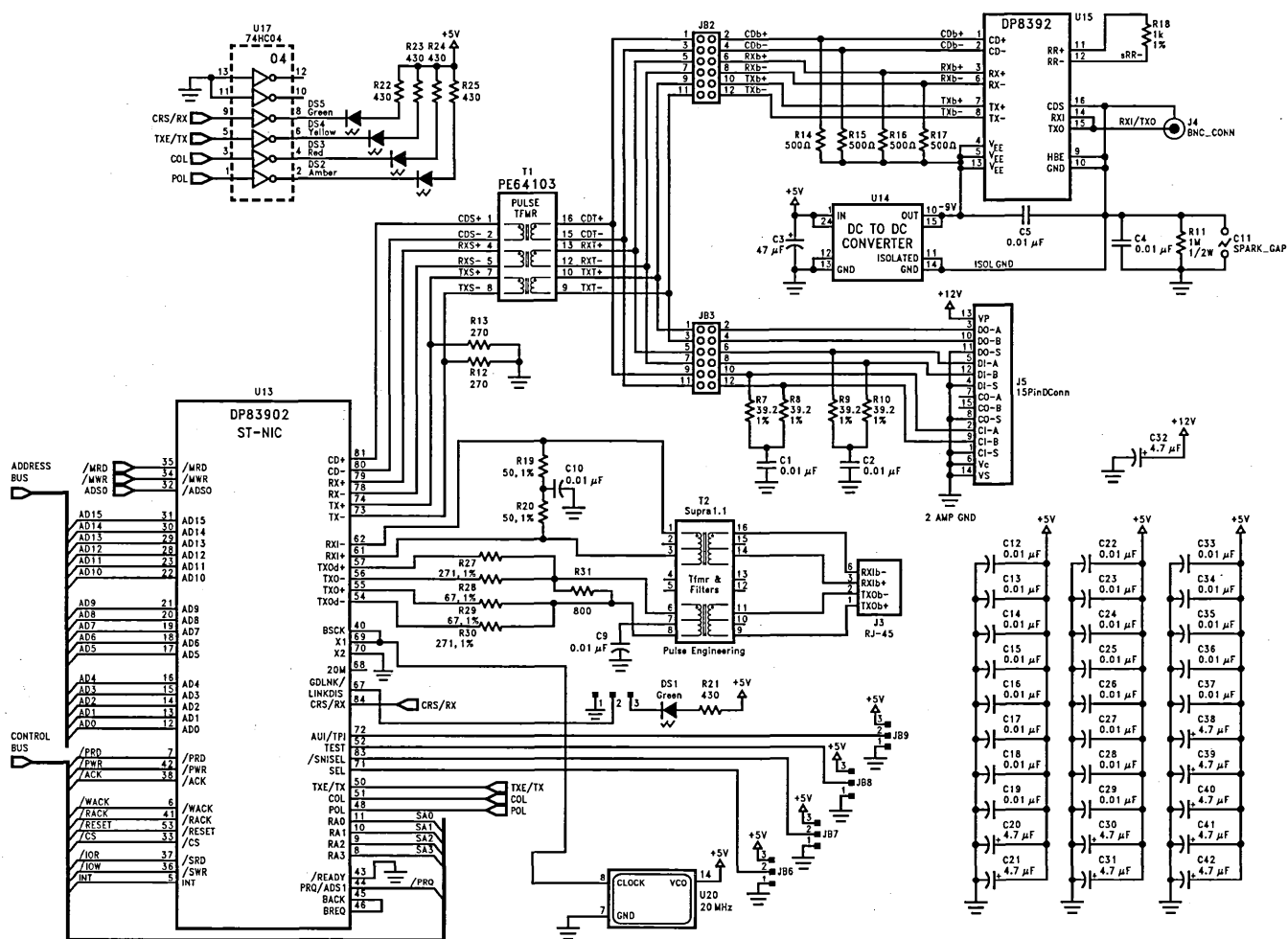
DS1 GREEN 5mm LOW CURRENT LED CURRENT= $I_F=3.5m$   
 DS2 AMBER 5mm LOW CURRENT LED CURRENT= $I_F=3.5$  mA  
 DS3 RED 5mm LOW CURRENT LED CURRENT= $I_F=2.0$  mA  
 DS4 YELLOW 5mm LOW CURRENT LED CURRENT= $I_F=2.0$  mA  
 DS5 GREEN 5mm LOW CURRENT LED CURRENT= $I_F=3.5$  mA  
 JB1 1x3 SHUNT BLOCK WITH .1" SPACING BETWEEN PINS  
 JB2 2x6 SHUNT BLOCK WITH .1" SPACING BETWEEN PINS  
 JB3 2x6 SHUNT BLOCK WITH .1" SPACING BETWEEN PINS  
 JB4 2x6 SHUNT BLOCK WITH .1" SPACING BETWEEN PINS  
 JB5-JB9 1x3 SHUNT BLOCK WITH .1" SPACING BETWEEN PINS

**SOCKETS/MECHANICAL**

S1-S3 20 PIN 0.3" DUAL IN-LINE FOR U1, U2, U16 (PAL)  
 S4 28 PIN DUAL IN-LINE SOCKET FOR U18 (EPROM)  
 S5 84 PIN PLCC SOCKET FOR U13 (ST-NIC) AMP SOCKET  
 S8 BRACKET FOR MOUNTING IN PC-AT, SLOT  
 G44 Basic Blank,  
 J3 RJ-45 CONNECTOR AMP 520252-4  
 J4 BNC CONNECTOR RT/A Low Pro Amp 227161-7  
 J5 15 PIN D CONNECTOR Female AMP 747247-4 (or 747845-4)  
 MAXCON SUB D Slide Lock MDA 51220-1

**BOARD ATTACHMENT COMPONENTS**

- 1) Screw: Bind Head Slotted 4-40 x .250, Steel, (90277A106)
- 2) Washer: Lock Ext #4, Zinc/Steel, (91114A005)
- 3) Washer: Flat #4, Zinc-CRS, (90126A005)



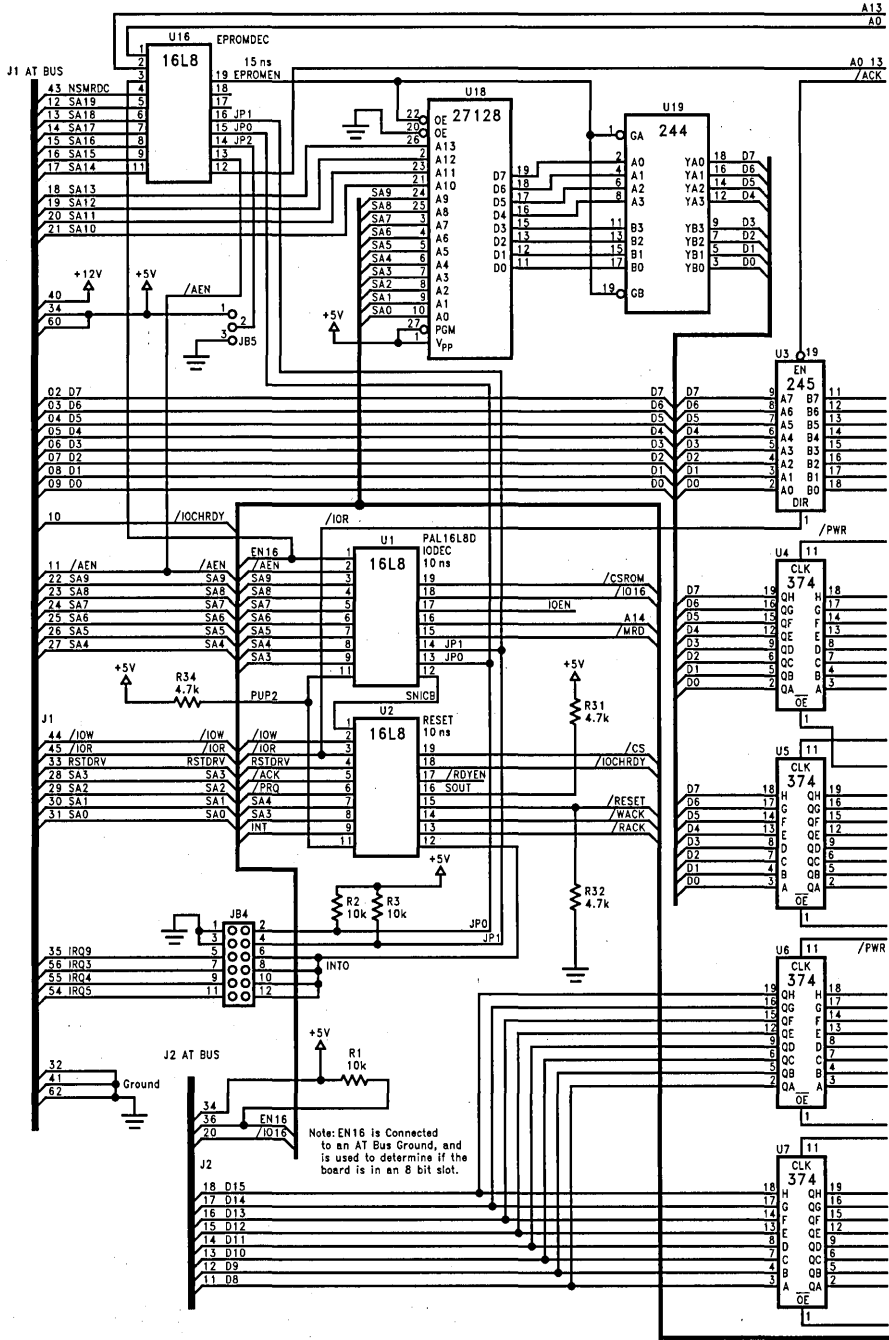
Note: All resistors to be 5%, 1/4W unless otherwise indicated  
 Note: Capacitors 39 to 42 are for the V<sub>CC</sub> signals off of the AT Bus.

DP83902EB-AT PC-AT Ethernet Evaluation Board

TL/F/11158-11

APPENDIX D

Bus Interface/NIC Section

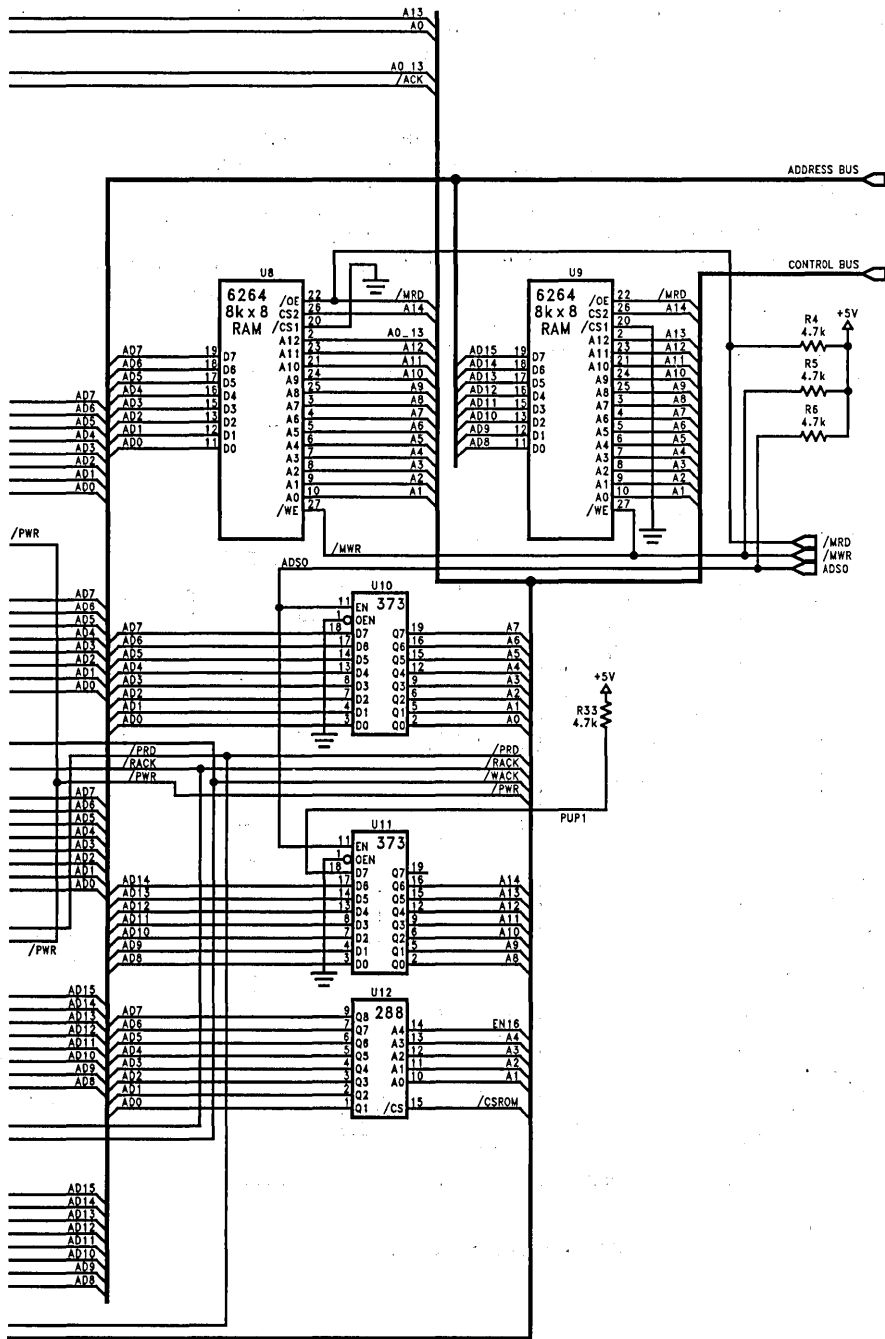


TL/F/11158-9

Note: All resistors to be 5%, 1/4W unless otherwise indicated

Note: EN16 is actually a ground signal on the AT Bus J2 Connector. This signal is used to determine whether 8- or 16-bit mode should be used.

DP83902EB-AT ST-NIC Ethernet Evaluation Board Schematic  
(Bus Interface/NIC Section)



DP83902EB-AT ST-NIC Ethernet Evaluation Board Schematic (Continued)  
(Bus Interface/NIC Section)

TL/F/11158-10

# NM95C12 Applications in a PC-AT® Ethernet® Adapter

National Semiconductor  
 Application Note 792  
 Sean Long



## INTRODUCTION

This application describes a typical Ethernet adapter card designed to be plugged into a PC-AT expansion slot. The board is designed around the National Semiconductor DP83932 SONIC™ Network Controller device. This application note will detail the system design and focus on the functions performed by the NM95C12 EEPROM.

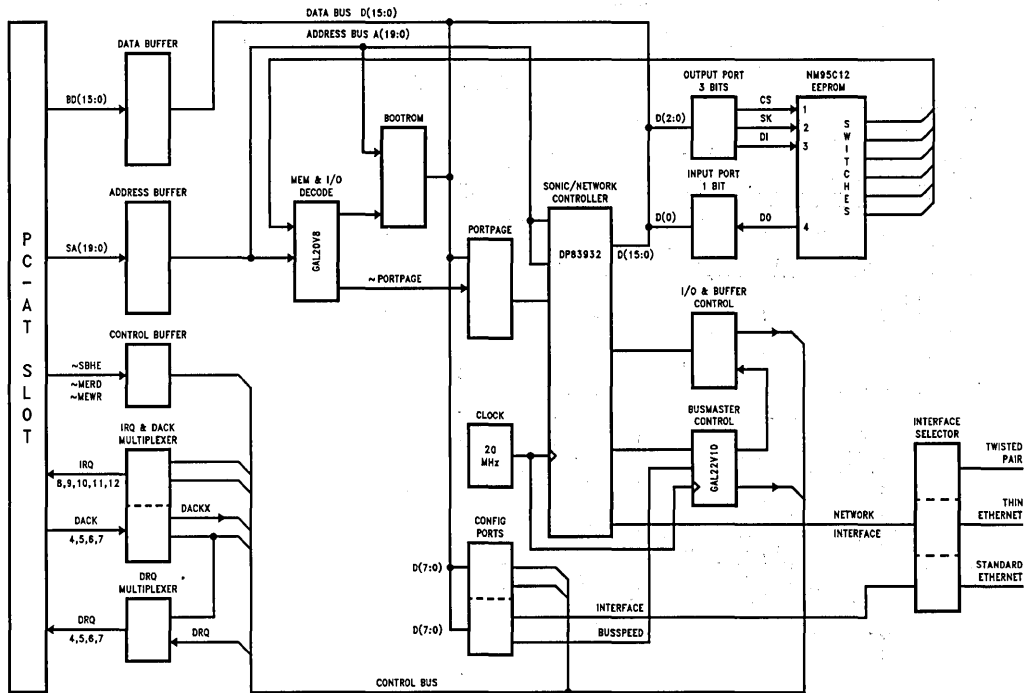
This application note assumes that the reader is familiar with the PC-AT architecture, the DP83932 device, the NM95C12 EEPROM and designing with GAL® Programmable Logic Devices (PLDs).

## SYSTEM DESCRIPTION

The network controller card has been designed to meet the following specifications:

- Designed around high performance 32-bit DP83932 Ethernet Controller
- 16-Bit bus master operation to give higher performance
- Fully software configurable (no jumpers or mechanical DIP switches)
- Extensive test and configuration capabilities
- Supports different media interfaces
- Bootrom option

The system block diagram is shown in *Figure 1*.



TL/D/11265-1

\*Denotes an active low signal.

FIGURE 1. System Block Diagram

## FUNCTIONAL DESCRIPTION OF THE BOARD

The system contains the following logical functions:

1. Network controller (DP83932)
2. Cable interfaces
3. Busmaster interface logic, including data and address buffers
4. EPROM option for remote boot loader

This system uses both the EEPROM locations and the switch logic terminals of the NM95C12 to perform various functions within the system as detailed below.

## FUNCTIONAL DESCRIPTION OF NM95C12 EEPROM

Use of the Switches:

The switch terminals of the NM95C12 EEPROM are used as part of the memory map address decoding and the I/O map decoding circuitry, feeding as inputs to a GAL20V8 which performs the address decoding logic from the system address inputs.

The NM95C12 switches control:

1. The base I/O address of the network controller board.
2. The base memory address of the bootrom EPROM option on the board.

## ADDRESS DECODING

The address decoding is controlled by a GAL20V8 PLD (refer to the 1990 National Semiconductor PLD Databook and Design Guide for further information) as shown in *Figure 2*.

The inputs to the GAL20V8 are the system address lines, the memory and I/O control signals, and the switch terminals

from the NM95C12. The outputs from the GAL20V8 are the various chip select signals for the memory and I/O ports. The system address bus transmits the current address value and the  $M/\sim IO$  signal determines if a memory or I/O cycle is in progress.

Address lines A0-A19 allow up to 1 Meg (0-FFFFFF) of memory to be addressed, while address lines A0-A15 allow up to 64K (0-FFFF) of I/O ports to be addressed. If the control signal  $M/\sim IO$  is logical "1" (high) then the processor is performing a memory cycle and if the  $M/\sim IO$  signal is logical "0" then an I/O cycle is in operation.

For a PC-AT various memory and I/O locations are reserved for standard functions such as system memory and I/O (refer to PC-AT documentation to determine which memory and I/O locations are free for add-in boards).

The switch outputs from the NM95C12 are connected as inputs to the GAL address decode logic and are used to determine the base memory and I/O locations for the add-in card. *Figure 2* shows the typical use of a GAL for address decoding.

The advantage of using a PLD for the address decoding is that it is an easy way to implement different address decode functions by logic equations. The logic equations can be implemented with a standard PLD design compiler such as OPAL™ from National Semiconductor or a third party software package such as ABEL™ from Data I/O. The PLD compiler will take the logic equations and convert them into the GAL fuse map which can be used for programming on a wide range of device programmers. A typical set of logic equations using National Semiconductors OPAL software package is shown in *Figure 3*.

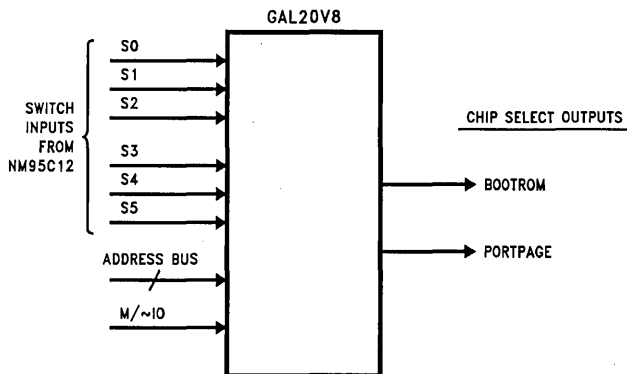


FIGURE 2. Address Decoding

TL/D/11265-2

```

BEGIN HEADER
TITLE   Address decoding for PC AT Ethernet adapter card
PATTERN Addr_Dec
REVISION Rev 0
AUTHOR   Dave Engineer
COMPANY  National Semiconductor
DATE     June 1991

Everything in the header command is copied directly into the JEDEC map as a comment field for
easy documentation
END HEADER

```

```

BEGIN DEFINITIONS
device G20V8;                { specify the device used }
inputs s0, s1, s2, s3, s4, s5;    { define the inputs }
inputs m_~io, a0, a1, a2, a3, a4, a5;
outputs (oom) bootrom, portpage;    {define the outputs}
{ OPAL will perform automatic pin assignment }
set ioselect=[s2,s1,s0], memselect=[s5,s4,s3]; { define the switch sets }
set address=[a5,a4,a3,a2,a1,a0];
END DEFINITIONS

```

```

BEGIN EQUATIONS
{ " / " = logical NOT function (i.e. logical 0)
  " & " = logical AND function
  " + " = logical OR function }
{ if m_~io is logical 0, then decode switch set s2, s1, s0 and address lines for the various
base I/O locations.
  Refer to PC-AT system I/O address map before selecting free I/O ports, the decodes shown are
  for example only - change for specific applications as required. }
bootrom = /m_~io & ( (ioselect == 0) & (address == ↑h00)
                    + (ioselect == 1) & (address == ↑h01)
                    + (ioselect == 2) & (address == ↑h02)
                    + (ioselect == 3) & (address == ↑h03)
                    + (ioselect == 4) & (address == ↑h04)
                    + (ioselect == 5) & (address == ↑h05)
                    + (ioselect == 6) & (address == ↑h06));
{ if m_~io is logical 1, then decode switch set s5, s4, s3 and address line for the various
base memory locations.
  Refer to PC-AT system I/O address map before selecting free Memory locations, the decodes
  shown are for example purposes only - change for specific applications as required. }
portpage = m_~io & ( (memselect == 0) & (address == ↑h18)
                    + (memselect == 1) & (address == ↑h20)
                    + (memselect == 2) & (address == ↑h28)
                    + (memselect == 3) & (address == ↑h30)
                    + (memselect == 4) & (address == ↑h38);
END EQUATIONS

```

FIGURE 3. GAL® Logic Equations

#### USE OF THE NM95C12 EEPROM LOCATIONS

- Three locations are used to store the ethernet address of the card.
- One location is used to store the interrupt number and the DMA channel of the board.
- One location is used to store the busmaster speed setting of the card.
- Two locations are used to store information about the production flow of the board e.g.; the version number of the out-going inspection, and serialization program which stores a unique ethernet address in the EEPROM.
- There are also some EEPROM locations used to enable some special features in the network driver such as protocol, DMA priority, etc.



Figure 4 below shows the memory usage of the NM95C12.

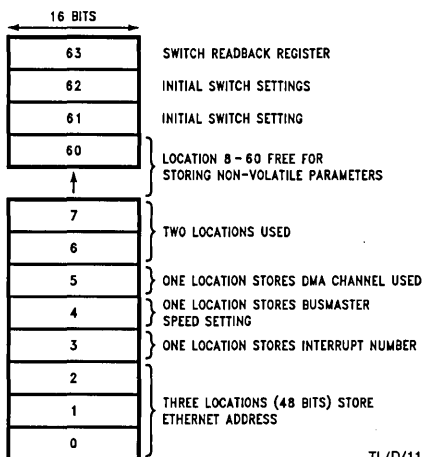


FIGURE 4. Memory Locations Used In NM95C12

#### FUNCTIONAL DESCRIPTION OF THE SOFTWARE

The driver for the card can be supplied in two ways:

1. As a driver which is loaded from the disk.
2. As a bootrom which is located at the card.

The driver determines the base I/O address of the card. This is done by scanning the possible I/O map where the card can be located (seven possible locations) and testing if the NM95C12 EEPROM can be found.

The EEPROM is found if, after an address is shifted in the EEPROM, the DO output from the NM95C12 has become logical "zero". Then the CS pin will be disabled and there will be a check if the DO output pin will become high (this pin is pulled up with a 47K resistor).

When the software finds the base address of the card, it reads the locations which contain the DMA and IRQ number to use and programs these values into the corresponding output latches. These latches will enable and/or multiplex the corresponding DMA (DACKx, DRQx) and INT (IRQx) to the busmaster logic and interrupt logic.

The same operation is done for the busmaster speed, one location in the EEPROM determines the active low and high time for busmaster cycles, the output of this latch will go to the busmaster state machine (implemented in a GAL22V10).

The ethernet address will be read by the driver and copied to a private location in the driver data area for use with the network software.

The bootrom can be located at five locations in memory (controlled by the NM95C12 switch logic) and can be disabled if required.

#### CONCLUSION

This application has shown the many advantages of the NM95C12 EEPROM with DIP Switches. In this example the NM95C12 replaces the functions typically performed by a Bipolar PROM (store ethernet address), mechanical DIP switches/jumpers (select options), and general read/write logic (software testing of the hardware configuration). The use of the switch terminals as part of the address decode logic makes the address decode function more flexible and allows for software control.

The easy interfacing to the NM95C12 (just four pins) and the simple, but powerful instruction set allows the NM95C12 to give the system designer:

- Greater flexibility
- Fully software controllable and testable
- Greater reliability (no mechanical switches or jumpers)
- Reduced component count
- Lower component cost

# The Design and Operation of a Low Cost, 8-Bit PC-XT® Compatible Ethernet Adapter Using the DP83902

National Semiconductor  
Application Note 842  
Larry Wakeman



## OVERVIEW

This 8-Bit Ethernet adapter board design is an inexpensive-low part count design that provides PC-XT and PC-AT® compatibles with Thick, Thin, and Twisted Pair Ethernet connectivity. This design provides an 8-bit interface that is compatible with Novell's 8-bit NE1000, while using the DP83902 (ST-NIC™) to interface to twisted pair Ethernet. The ST-NIC also has an AUI interface which allows interface to thick wire Ethernet, or thin wire Ethernet by the addition of the DP8392 Coaxial Transceiver Interface (CTI). The dual DMA (local and remote) capabilities of the ST-NIC, along with 8 kbytes of buffer RAM, and bus interface logic provide a high performance 8-bit interface. The I/O port architecture used in this design isolates the CPU from the network traffic, proves to be the simplest method to interface the DP83902 to a PC system bus.

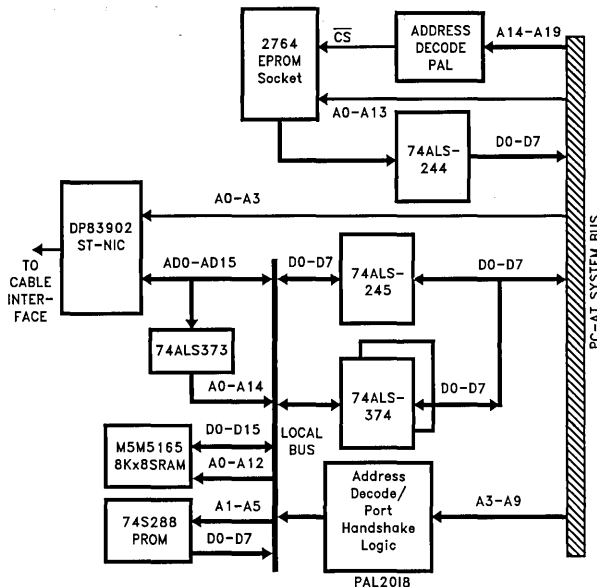
This paper describes the basic design and operation of this 8-bit adapter card, and then follows this with specific design information including the PAL equations and a detailed schematic of the design. For detailed information refer to the schematics at the end of this document.

## HARDWARE FEATURES

- Fits in a half-size IBM XT form factor
- Utilizes DP83902 twisted pair network interface controller (ST-NIC)
- 8 kbyte on-board packet buffer
- Simple I/O port interface software compatible with Novell's NE1000 Ethernet adapter
- Interfaces to Thick (10BASE5), Thin (10BASE2), and Twisted Pair (10BASE-T) Ethernet
- Boot EPROM socket

## BUS INTERFACE

The block diagram for this design is shown in *Figure 1*. The ST-NIC board as seen by the system appears only to be a block of I/O ports. With this architecture the ST-NIC board has its own local bus to access the board's memory. The system never has to intrude further than the I/O ports for any packet data operation. There are two register/memory maps that describe the card. The first is the I/O register map that describes how the PC's processor accesses the card. The second map is the one that describes how the ST-NIC accesses the on-card memory.



TL/F/11492-1

FIGURE 1. System Bus Block Diagram for 8-Bit Ethernet Card

**I/O Map**

The ST-NIC board requires a 32 byte I/O space to allow for decoding the data port, the reset port, and the ST-NIC registers. This is shown in Table I. The first 16 bytes are the ST-NIC registers, the next 16 bytes address the data port and the reset port, which are aliased alternately as shown.

**TABLE I. I/O Map In PC-AT**

Address Offset	Device Accessed
00h-0Fh	ST-NIC Registers
10h-13h	Data I/O Port
14h-17h	Reset Port
18h-1Bh	Data I/O Port
1Ch-1Fh	Reset Port

Additionally there are three jumpers which define the base addresses for the address map shown in Table I. (See the Jumper Configuration section for details.)

**On-Card Memory Map**

There are only two items mapped into the local memory space. These two items being the 8k x 8 buffer RAM and the Ethernet ID address PROM. The buffer RAM is used for temporary storage of transmit and receive packets.

**TABLE II. ST-NIC's Local Memory Map**

	etc.	ST-NIC A13 Low = PROM ST-NIC A13 High = RAM
7FFF	RAM	← The PROM and RAM are Aliased through 64K Address Range.
6000 5FFF		
4000 3FFF	PROM	
2000 1FFF	RAM	
0000	PROM	← The 'S288 PROM Actually Takes 32 Locations and is Aliased for Each 8K Block.

The buffer RAM is used for temporary storage of network packet data that is either being transmitted or received. The ID address PROM (74S288 32 x 8) contains the physical address of the evaluation board. Each PROM holds its own unique physical address which is installed during its manufacture. Besides this address, the PROM also contains some identification bytes that can be checked by the driver software. At the initialization of the evaluation board the software commands the ST-NIC to transfer the PROM data

to the I/O Port where it is read by the CPU. The CPU then loads the ST-NIC's physical address registers. The following table shows the contents of the PROM.

**TABLE III. PROM Contents**

PROM Location	Location Contents
00h	Ethernet Address 0 (Most Significant Byte)
01h	Ethernet Address 1
02h	Ethernet Address 2
03h	Ethernet Address 3
04h	Ethernet Address 4
05h	Ethernet Address 5
06h-0Dh	00h
0Eh, 0Fh	57h
10h-15h	Ethernet Address 0-5
16h-1Dh	Reserved
1Eh, 1Fh	42h

**EPROM INTERFACE**

An EPROM socket is provided so that the end user may add an EPROM to the system. This EPROM would normally contain a program and a driver to enable the PC-AT to be boot up (Operating System loaded) from a designated network server. The ICs necessary to interface the EPROM to a 16L8 (PAL), and a 74ALS244 (buffer). The PAL decodes SA14-SA19, along with system memory read, in order to generate the EPROM enable signal. The '244 provides buffering of the EPROM's data bus to the PC's bus.

**General Bus Interface Operation**

For receiving of packets, the ST-NIC first starts to receive a packet and checks the address of this packet. If the address corresponds to the address for this card, then the data is received by the ST-NIC. The ST-NIC utilizes it's Local DMA channel to buffer the packet into the next available area of the 8k buffer RAM. As each packet is received the ST-NIC's local DMA will buffer it to the next available location in memory. After each packet is buffered the ST-NIC will generate and interrupt to the CPU. If a packet that has an error is loaded into RAM, the ST-NIC will reject the packet and reclaim the memory space that the packet occupied.

Upon recognition of the receive interrupt the CPU should then program the ST-NIC's Remote DMA to read the packet into the I/O port consisting of the two back-to-back 74ALS374s (see Figure 1). As the ST-NIC's Remote DMA does this the CPU handshakes with the DMA to read each byte from the I/O Data Port and store it in the PC's main memory. This is repeated until the packet has been completely transferred.

For packet transmission, the system CPU first programs the ST-NIC's Remote DMA to receive a packet from the system into a predetermined area of the card's buffer RAM. The CPU and the ST-NIC then handshake while the data is sent through the I/O data port.

Once the transmit packet is completely assembled into the local card RAM, the ST-NIC is then programmed to transmit the packet out onto the network. The ST-NIC then reads the transmit data using its Local DMA channel, and then follows the CSMA/CD protocol to transmit the data. When the transmission is complete an interrupt is generated, and the CPU can check the status of the transmit to ensure proper transmission did occur.

**NETWORK INTERFACE**

The evaluation board supports three physical media interfaces options: Thick Ethernet, Thin Ethernet, and Twisted Pair. The block diagram for these interfaces can be seen in *Figure 2*. A single jumper selects between the ST-NIC's Attachment Unit Interface (AUI) and its 10BASE-T interface. When the AUI is selected a second jumper selects the Thin Ethernet interface or the AUI connector. This second jumper shorts/opens the power supply to the transceiver. The AUI interface provides connectivity to an external transceiver which typically connects to Thick Ethernet cable.

The ST-NIC has an integrated 10BASE-T interface so that all that needs to be added are the equalization resistors, and an integrated filter module such as the Valor FL1012.

The Thin Ethernet interface is a little more complicated. This section includes a pulse transformer and a DC-DC Converter (Valor PM7102 or equivalent) to provide the required isolation, and the DP8392 Coax Interface and a few discrete components. The input power to the PM7102 is enabled via the jumper to enable disabling of this interface.

**JUMPER CONFIGURATIONS**

On the DP83902EB-AT ST-NIC AT board, there are nine jumpers as grouped in the one block in the component layout shown in *Figure 3*. The following pages will explain how to configure these jumpers, and what they do.

**Physical Interface**

There are a number of jumper options provided in this design to enable utilizing some of the ST-NIC's pin programmable options. Most of these are set for the normal default and should not have to be changed (in fact most are only provided for experimentation purposes). These jumpers are JP3, JP5, and JP6. JP4 is provided to enable experimentation with different bus clocks should a designer wish. This option is not useful for general operation and the default to use the 20 MHz network clock would normally be used.

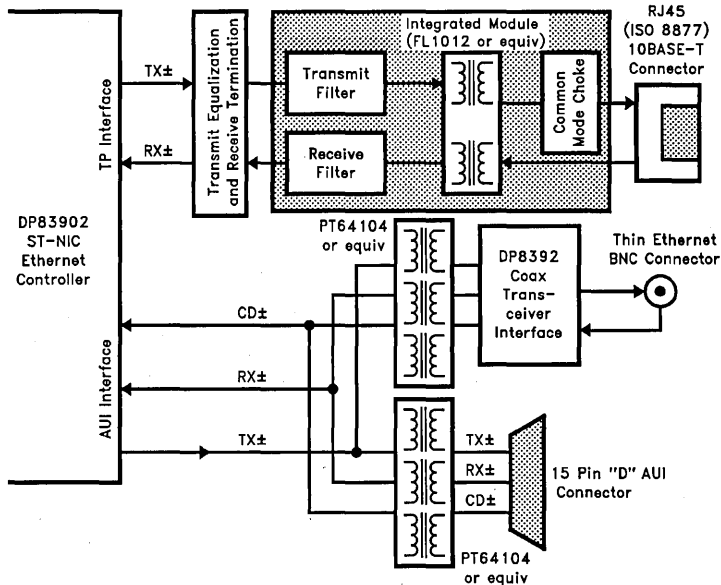
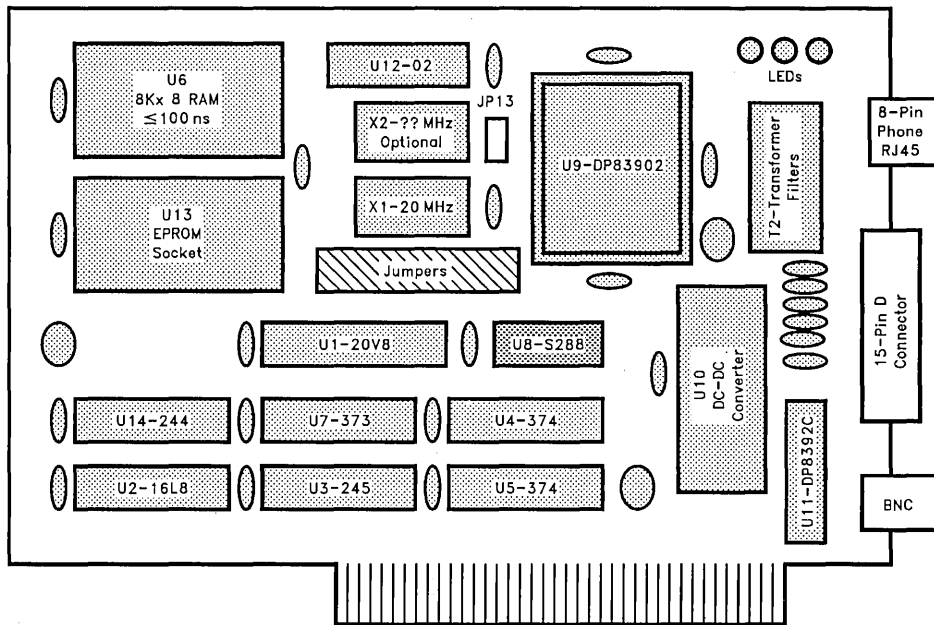


FIGURE 2. Media Interface Block Diagram

TL/F/11492-2

Proposed Component Placement



TL/F/11492-3

FIGURE 3. Proposed Component Placement (Jumpers Located in Center of Board)

TABLE IV. ST-NIC Option Jumpers

Jumper	Position	Description	
JP3	OFF	IEEE Half Step AUI Mode	Default
	ON	Ethernet Full Step AUI Mode	
JP4	ON	Common Network-Bus Clocks	Default
	OFF	Separate Network-Bus Clocks	
JP5	ON	Link LED Off	Default
	OFF	LED Enabled	
JP6	OFF	10BASE-T Link Enabled	Default
	ON	10BASE-T Link Disabled	

Two jumpers select which physical media to use as shown in Table V.

TABLE V. Physical Media Selection

JP8	JP7	Description	
OFF	OFF	Thick Coax (10BASE5)	Default
OFF	ON	Twisted Pair (10BASE-T)	
ON	OFF	Thin Coax (10BASE2)	
ON	ON	Illegal	

For Table V, the selection default is Thin Ethernet, and any of the jumper options may be selected, except shorting both jumpers, JP7 and JP8. If this is done then the Thinnet transceiver is enabled, but the ST-NIC will use the twisted pair interface, since this does not make any sense the option is not useable.

I/O and EPROM Addresses

This design utilizes the same set of I/O address selections and EPROM address selections as NE1000, as shown in Table VI.

TABLE VI. I/O and EPROM Address Options

JP2	JP1	JP0	I/O Address	EPROM Address	
ON	ON	ON	300H	C800H	Default
ON	ON	OFF	300H	Disabled	
ON	OFF	ON	320H	CC00H	
ON	OFF	OFF	320H	Disabled	
OFF	ON	ON	340H	D000H	
OFF	ON	OFF	340H	Disabled	
OFF	OFF	ON	360H	D400H	
OFF	OFF	OFF	360H	Disabled	

As can be seen JP0 actually is the enable for the EPROM, and JP1 and JP2 select the addresses for both the I/O and EPROM. Like the NE1000 the addressing of the I/O and EPROM cannot be set individually.

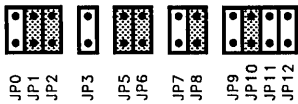
This design supports the same interrupt selection options as the NE1000, as shown in Table VII. Individual jumpers enable each interrupt to the bus interface. It is important that only one interrupt be selected at any one time.

**TABLE VII. Interrupt Output Selection**

JP12	JP11	JP10	JP9	Interrupt
ON	OFF	OFF	OFF	IRQ6
OFF	ON	OFF	OFF	IRQ4
OFF	OFF	ON	OFF	IRQ3
OFF	OFF	OFF	ON	IRQ9
OFF	OFF	OFF	OFF	NONE

Default

All of these jumper options can be provided in a relatively easy grouping as shown below. (JP5) may not be grouped here as it is better to place it near the ST-NIC to try to minimize clock trace lengths.



TL/F/11492-4

**FIGURE 4. Possible Jumper Configuration**

**LAYOUT CONSIDERATIONS**

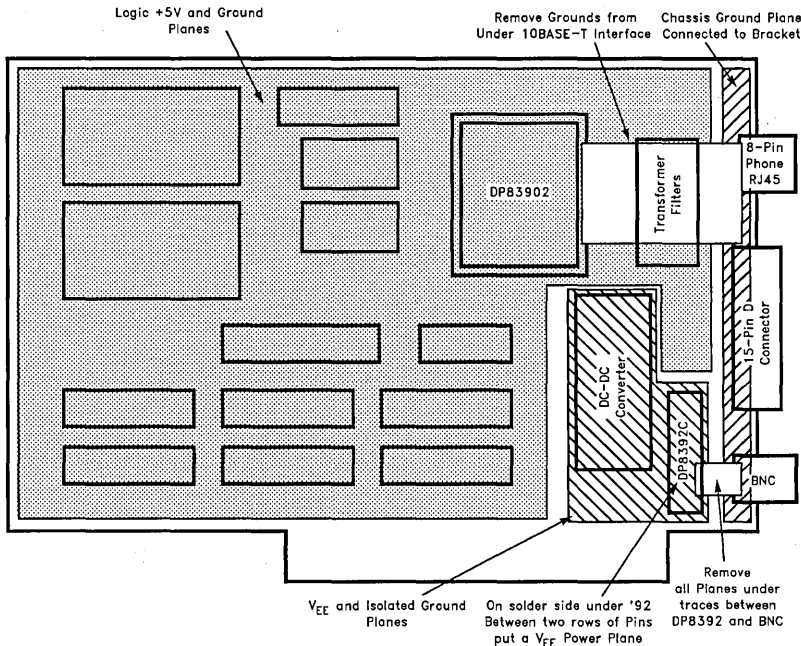
The PCB layout for this design is very similar to most triple media interface designs (most of the layout considerations

revolve around the media interfaces layout). The major component placement decisions are to place the ST-NIC's 10BASE-T port near the RJ45 Connector, and to place the DP8392 close to the BNC connector.

For the ST-NIC placement and layout, it is important to ensure that the power supply noise imparted from the board is minimized. To ensure this adequate decoupling around the 4 sides of the ST-NIC is important. There are two reasons for this. First the ST-NIC is a combined digital and analog function so to maximize the analog circuit performance, noise should be reduced. Secondly, the AUI and twisted pair outputs can conduct power supply noise out to the connectors. Thus power supply noise should be kept to a minimum to reduce RFI emissions. It is recommended that 0.1  $\mu$ F low ESR decoupling capacitors be used along with a couple of 4.7  $\mu$ F-10  $\mu$ F tantalum capacitors.

On the ST-NIC's twisted pair interface, the layout should be compact, and all signal traces should be kept straight and short. It is preferable to have each of the signals in a particular differential pair matched to minimize differential skews (i.e., RX+ and RX- should be matched). Also, the power planes under the twisted pair interface components should be removed to prevent power supply noise from being injected into the twisted pair signals, again to minimize RFI.

For the DP8392 layout there are several considerations. First the CTI power planes must be isolated from the logic power planes by a PCB gap that can withstand 500V. The isolated power plane should be removed from under the signals that interface from the CTI to the BNC connector. This is required to reduce the capacitance as seen from the Thin net (RG58) cable. It is also advisable to add a small heatsink power plane to the solder side layer that encompasses the area between the two rows of pins of the DP8392 package (see datasheet for specific layout recommendations).



**FIGURE 5. Ground/Power Planes and Layout Considerations**

TL/F/11492-5

## BILL OF MATERIALS July 8, 1991

CAPACITORS	
C1, C2, C6-C26, C31-C35	0.01 $\mu$ F, 50V Monolythic
C3, C27, C28, C36	22 $\mu$ F, 12V Tantalum 20%
C4	0.01 $\mu$ F, 1 kV Ceramic
C5	0.01 $\mu$ F, 50V Ceramic
C29, C30	0.01 $\mu$ F, 50V Ceramic
RESISTORS (5% $\frac{1}{8}$ Watt unless otherwise noted.)	
R1-R3	4.7k
R4, R5	270, $\frac{1}{4}$ W
R6-R9	39.2, 1%
R10-R13	1.5k
R14	1k, 1%
R15	150, 1%, $\frac{1}{4}$ W
R17	1M, $\frac{1}{2}$ W
R18	10k, 1%
R19, R20	50, 1%
R22, R23	66, 1%, $\frac{1}{4}$ W
R21, R24	271, 1%
R25	800, 1%
R26-R28	300
R29-R35	4.7k
DIODES	
D1	1N4150 (FDSO1201 SMT Version)
D2	Green LED 5mm Low Current
D3	Amber LED 5mm Low Current
D4	Green LED 5mm Low Current
CONNECTORS/SOCKETS	
J2	BNC Same as ATT
J3	RJ-45 AMP 520252 or Non-Keyed or Equivalent
J4	15-Pin D Conn Female, 747247-4 Slide Lock AMD MDA 51220-1
JP0-JP13	2 Pin Jumper, 0.1" Pin Space

CONNECTORS/SOCKETS (Continued)	
S1	24-Pin 0.3" Space Socket for U1
S2	20-Pin 0.3" Space Socket for U2
S3	16-Pin 0.3" Space Socket for U8
S4	24-Pin 0.3" Space Socket for U13
SEMICONDUCTORS	
U1	GAL20V8-15 or PAL20L8 (Socketed)
U2	PAL16L8-15 (Socketed)
U3	74ALS245
U4, U5	74ALS374
U6	HME264-85ns (May Use 100ns)
U7	74F373
U8	74S288
U9	DP83902
U11	DP8392C
U12	74ALS02
U13	27128 (Socket Installed Only)
U14	74ALS244
MISC	
SP1	0.75 pF, 1 kV, Spark Gap Mallory ASR75A or MEPCO/CENTRALAB S758X44000NAZAA
T1	Belfuse S553-1006-AE
T2	Supra1.1 10BASE-T Pulse Transformer/Filter
U10	PM7102 Valor DC-DC
X1	20 MHz, 0.01%, Oscillator 40/60% 10 TTL Drive Opt.
X2	25 MHz, 0.1%, Oscillator 40/60% (Not Installed) Opt.
	Bracket for Mounting in PC-AT Slot G44 Basic Blank, Stamped as DP839EB-ATS Board (Assy. # 980550173) Screw: Bind Head Slotted 4—40 x 0.250, Steel, (90277A106) Washer: Lock Ext #4, Zinc/Steel, (91114A005) Washer: Flat #4, Zinc-CRS, (90126A005)

**PAL EQUATIONS**  
**PAL #1 (U1)**

```

module iodecode    flag '-r1'
title 'date: 7/5/91
functions: IO Address Decode, IO Port-NIC Handshake, Ready Generation'

ul device 'P20L8';

"input pins:
BCLK, SA9, SA8, SA7, nJPEN           pin 1, 2, 3, 4, 5;
SA4, SA2, RSTD, nIOR, nIOW          pin 6, 7, 8, 9, 10;
PRQ, nAEN, nACK                     pin 11, 14, 23;

"output pins:
nIORDY, nCSN, nWACK, nIOEN          pin 15, 17, 18, 19;
nNRST, NRST, nRACK                  pin 20, 21, 22;

"constants
    X, Z, H, L = .X..Z., 1, 0;
    ADDR2 = [SA9, SA8, SA7, X, X, SA4, X, SA2, X, X];

equations

nCSN = (((!nAEN & !nJPEN & !SA4 & SA9 & SA8 & !SA7 & !nIOR)
        # (!nAEN & !nJPEN & !SA4 & SA9 & SA8 & !SA7 & !nIOW));

nRACK = (!(nAEN & !nJPEN & SA9 & SA8 & !SA7 & SA4 & !SA2 & !nIOR & PRQ);

nWACK = (!(nAEN & !nJPEN & SA9 & SA8 & !SA7 & SA4 & !SA2 & PRQ & !nIOW);

nIOEN = (((!nAEN & !nJPEN & !nIOR & SA9 & SA8 & !SA7 & SA4 & !SA2) "Not Reset
        # (!nAEN & !nJPEN & !nIOW & SA9 & SA8 & !SA7 & SA4 & !SA2)
        # (!nAEN & !nJPEN & SA9 & SA8 & !SA7 & !SA4 & !nIOR) "NIC Registers
        # (!nAEN & !nJPEN & SA9 & SA8 & !SA7 & !SA4 & !nIOW));

enable nIORDY = !nIOEN;
nIORDY = !(nACK & !nCSN
        # !PRQ & nCSN);

NRST = !(nIOW # nNRST);
nNRST = (((!nIOR & !nAEN & !nJPEN & SA9 & SA8 & !SA7 & SA4 & SA2)
        # RSTD # NRST);

test_vectors ([ADDR2, nAEN, nIOR, nIOW, nJPEN] -> [nCSN]);

" nCSN ASSERTION

"  A  nA nI nI nJP  nC
"  D  E  O  O  E  S
"  D  N  R  W  N  N

[^h310, L, L, L, L] -> [H]; " None
[^h310, L, H, H, L] -> [H]; " None
[^h300, H, L, L, H] -> [H]; " None
[^h300, L, L, L, H] -> [H]; " None
[^h300, L, L, L, L] -> [L]; " nCSN
[^h300, L, L, H, L] -> [L]; " nCSN

```



```

[^h308, L, H, L, L] -> [L]; " nCSN
[^h320, L, L, L, H] -> [H]; " None

test_vectors ([ADDR2, nAEN, nIOR, nIOW, nJPEN, PRQ] -> [nRACK, nWACK]);

" A nA nI nI nJP P nR nW
" D E O O E R C C
" D N R W N Q K K

[^h310, L, L, H, L, H] -> [L, H]; " nRACK
[^h310, L, H, L, L, H] -> [H, L]; " nWACK
[^h310, H, L, H, H, H] -> [H, H]; " None
[^h310, L, L, H, H, H] -> [H, H]; " None
[^h310, H, L, L, L, H] -> [H, H]; " None
[^h310, H, L, L, H, H] -> [H, H]; " None
[^h300, L, L, L, H, H] -> [H, H]; " None
[^h314, L, L, L, H, L] -> [H, H]; " None
[^h318, L, L, H, L, H] -> [L, H]; " nRACK
[^h318, L, H, L, L, H] -> [H, L]; " nWACK

test_vectors ([ADDR2, nAEN, nIOR, nIOW, PRQ, nACK, nJPEN] -> [nIORDY]);

" A nA nI nI P nA nJP nI
" D E O O R C E R
" D N R W Q K N DY

[^h300, L, L, H, X, H, L] -> [L]; " NIC Read
[^h300, L, L, H, X, L, L] -> [H]; " NIC Read Ready
[^h300, L, H, L, X, H, L] -> [L]; " NIC Write
[^h300, L, H, L, X, L, L] -> [H]; " NIC Write Ready

[^h310, L, L, H, L, X, L] -> [L]; " IO Read
[^h310, L, L, H, H, X, L] -> [H]; " IO Read Ready
[^h310, L, H, L, L, X, L] -> [L]; " IO Write
[^h310, L, H, L, H, X, L] -> [H]; " IO Write Ready

test_vectors ([ADDR2, nAEN, nIOR, nIOW, RSTD, nJPEN] -> [nNRST, NRST]);

" A nA nI nI R nJP nN N
" D E O O S E R R
" D N R W T N T T

[^h300, H, H, H, H, L] -> [L, H]; " Hard Reset
[^h300, H, H, H, L, L] -> [L, H]; " Reset Latched
[^h300, H, H, L, L, L] -> [H, L]; " Un Reset
[^h314, L, L, H, L, L] -> [L, H]; " Soft Reset
[^h314, L, H, H, L, L] -> [L, H]; " Reset Latched
[^h300, H, H, L, L, L] -> [H, L]; " Un Reset
[^h30C, L, L, H, L, L] -> [H, L]; " Soft Reset
[^h30C, L, H, H, L, L] -> [H, L]; " Reset Latched
[^h300, H, H, L, L, L] -> [H, L]; " Un Reset

end iodecode;

```

## PAL #2

```

module epdecode flag '-r1';
title '
date:7/5/91
functions: EPROM DECODE, ID PROM DECODE, INTERRUPT BUFFER'

u2 device 'P16L8';

"input pins:
nAEN, nMEMR, SA19, SA18          pin 1, 2, 3, 4;
SA17, SA16, SA15, SA14          pin 5, 6, 7, 8;
SA13, A5, A6                    pin 9, 11, 13;
NINT, JP2, JP1, JP0             pin 14, 15, 16, 17;

"output pins:
INT, nJPEN, nCSEP               pin 12, 18, 19;

"constants
X, Z, H, L = .X...Z., 1, 0;
ADDR = [SA19, SA18, SA17, SA16, SA15, SA14, SA13, X, X, X, X, X, X, , X, X];

equations
nCSEP = (!((!nAEN & !nMEMR & !JP2 & !JP1 & !JP0 & (ADDR == ^hc800))
# (!nAEN & !nMEMR & !JP2 & JP1 & !JP0 & (ADDR == ^hcc00))
# (!nAEN & !nMEMR & JP2 & !JP1 & !JP0 & (ADDR == ^hd000))
# (!nAEN & !nMEMR & JP2 & JP1 & !JP0 & (ADDR == ^hd400)));

nJPEN = (!((!JP2 & !JP1 & !A5 & !A6)      " 300H
# (!JP2 & JP1 & A5 & !A6)                " 320H
# ( JP2 & !JP1 & !A5 & A6)                " 340H
# ( JP2 & JP1 & A5 & A6));                " 360H

INT = (!(NINT));

test_vectors ([ADDR, nAEN, nMEMR, JP2, JP1, JP0] -> [nCSEP]);

" A   A M J J J   C
" D   E M P P P   S
" D   N R 2 1 0   EP

[^hc800, L, L, L, L, L] -> [L]; " Proper Decode
[^hcc00, L, L, L, H, L] -> [L]; "
[^hd000, L, L, H, L, L] -> [L]; "
[^hd400, L, L, H, H, L] -> [L]; "
[^hc800, L, L, L, L, H] -> [H]; " Jumper Disable
[^h0000, L, L, L, L, L] -> [H]; " No Address
[^hc800, L, H, L, L, L] -> [H]; " No MRD
[^hc800, H, L, L, L, L] -> [H]; " No AEN
[^hFF00, L, L, L, L, L] -> [H]; " No Address
[^hc800, L, L, H, L, L] -> [H]; " No JP2
[^hcc00, L, L, L, L, L] -> [H]; " No Address

test_vectors ([JP2, JP1, A5, A6] -> [nJPEN])

" J J A A   J
" P P 5 6   P
" 2 1      EN

[H, L, L, L] -> [H]; " No Enable
[L, H, L, L] -> [H]; " No Enable

```

```

[L, L, H, L] -> [H]; " No Enable
[L, L, L, H] -> [H]; " No Enable
[L, L, L, L] -> [L]; " Enable
[H, L, L, H] -> [L]; " Enable
[L, H, H, L] -> [L]; " Enable
[H, H, H, H] -> [L]; " Enable

```

```
test_vectors ({NINT} -> {INT});
```

```

[H] -> [H]; "Non Inverter
[L] -> [L];

```

```
end epdecode;
```

TL/F/11492-9

**ETHERNET ADDRESS PROM CONTENTS**

The ID PROM is 74S288 type. The content is as follows:

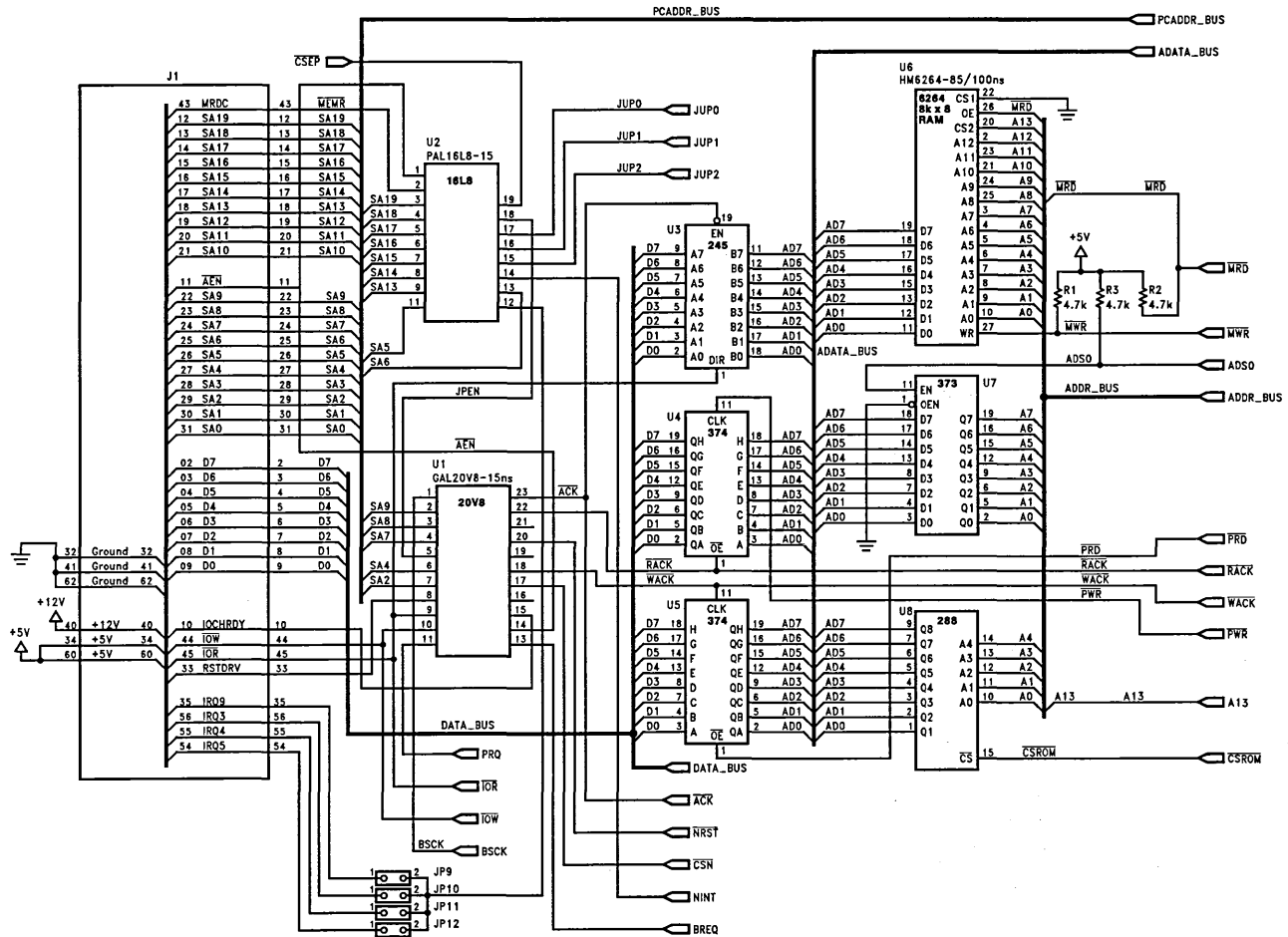
```

ADDR 00:  08  00  17  xx  yy  zz  00  00  00  00  00  00  00  00  00  57  57
ADDR 10:  08  00  17  xx  yy  zz  00  00  00  00  00  00  00  00  00  42  42

```

The ID address is 080017xyyzz where 080017 is National's ID "prefix".

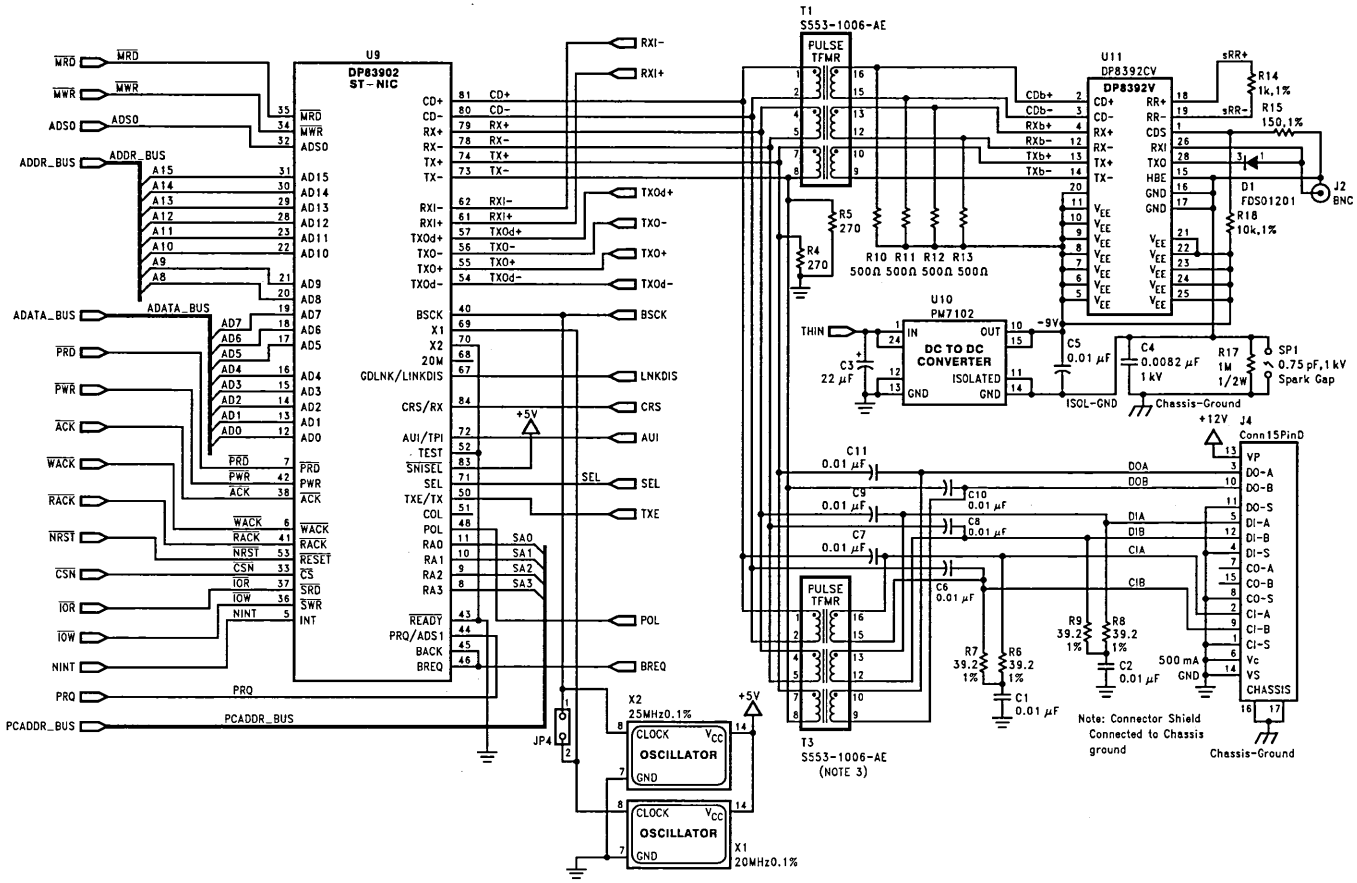
FIGURE 6. DP83902EB-XT 8-Bit Ethernet Adapter with 10BASE-T



1-414

Note: All resistors to be 5%, 1/4W, unless otherwise indicated.

FIGURE 6. DP83902EB-XT 8-Bit Ethernet Adapter with 10BASE-T (Continued)

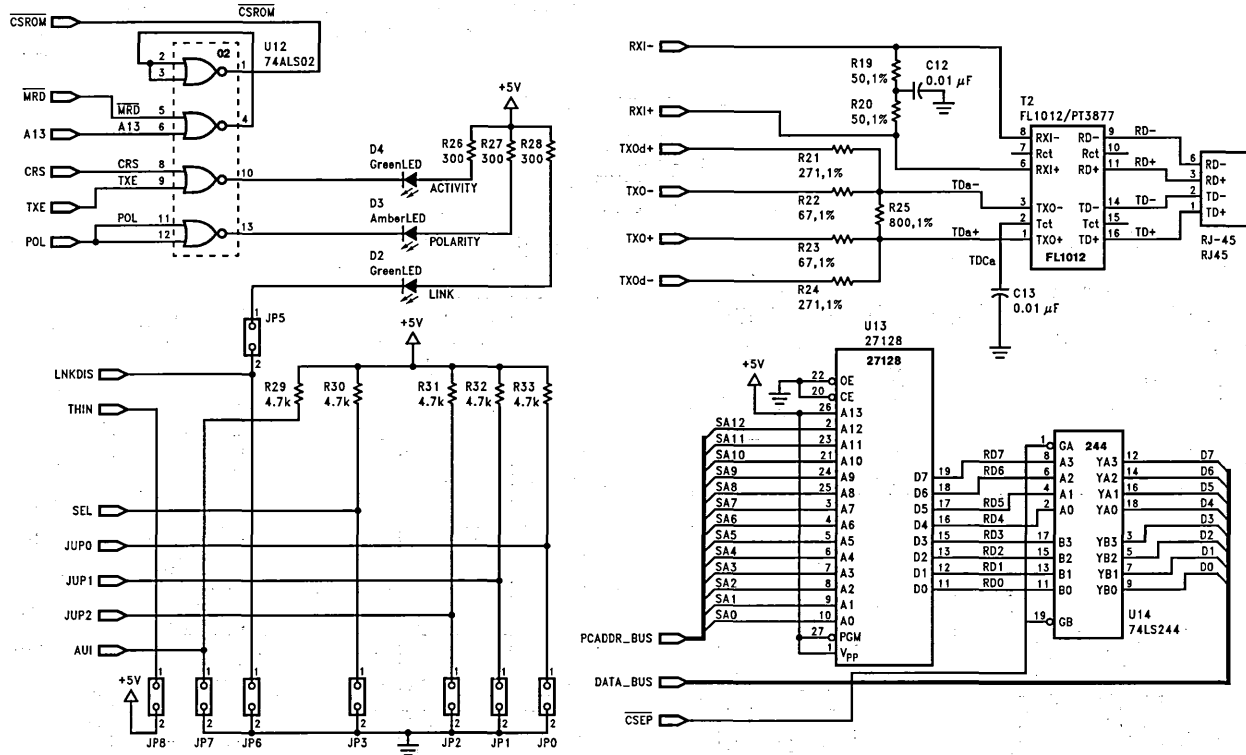


- Note 1:** All resistors to be 5%, 1/4W, unless otherwise indicated.
- Note 2:** X2 is not populated but the location is provided to provide optional bus clock speeds.
- Note 3:** T3 and C6-C11 are not populated simultaneously. Either T3 can be populated or the capacitors can be installed.

TL/F/11492-11

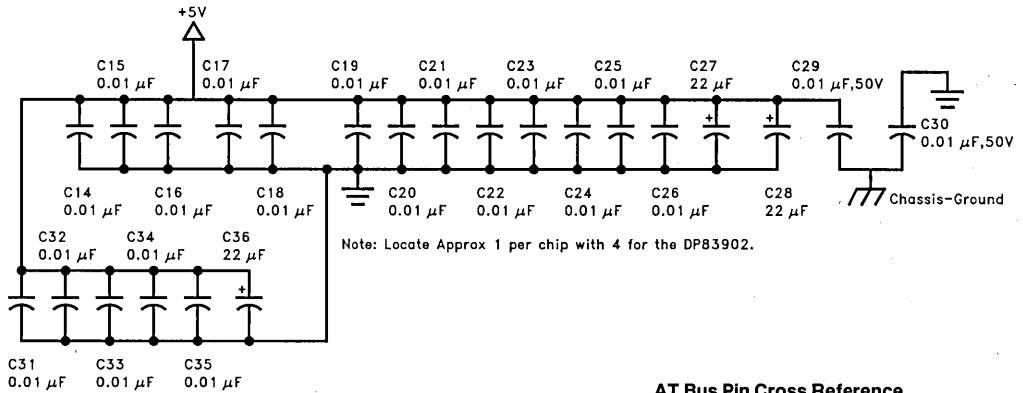
1-415

FIGURE 6. DP83902EB-XT 8-Bit Ethernet Adapter with 10BASE-T (Continued)



1-416

FIGURE 6. DP83902EB-XT 8 Bit Ethernet Adapter with 10BASE-T (Continued)



TL/F/11492-13

AT Bus Pin Cross Reference

AT Pins	SCH No.	AT Pins	SCH No.	AT Pins	SCH No.
A2	02	A23	23	B12	43
A3	03	A24	24	B13	44
A4	04	A25	25	B14	45
A5	05	A26	26	B15	46
A6	06	A27	27	B16	47
A7	07	A28	28	B17	48
A8	08	A29	29	B18	49
A9	09	A30	30	B19	50
A10	10	A31	31	B20	51
A11	11	B1	32	B21	52
A12	12	B2	33	B22	53
A13	13	B3	34	B23	54
A14	14	B4	35	B24	55
A15	15	B5	36	B25	56
A16	16	B6	37	B26	57
A17	17	B7	38	B27	58
A18	18	B8	39	B28	59
A19	19	B9	40	B29	60
A20	20	B10	41	B30	61
A21	21	B11	42	B31	62
A22	22				

# Guide to Loopback Using the DP8390 Chip Set

National Semiconductor  
Application Note 858



## OVERVIEW

Loopback capabilities are provided to allow certain tests to be performed to validate operation of the DP8390 NIC, the DP8391 SNI, and the DP8392 CTI prior to transmitting and receiving packets on a live network. Typically these tests may be performed during power up of a node. The diagnostic provides support to verify the following:

1. Verify integrity of data path through each chip. Received data is checked against transmitted data.
2. Verify CRC logic's capability to generate good CRC on transmit.
3. Verify CRC recognition capability of the NIC on receive.
4. Verify that the address recognition logic can
  - a. Recognize address match packets
  - b. Reject packets that fail to match an address

## LOOPBACK MODES

Loopback modes are selected by programming the Transmit Configuration Register. Bits LB0 and LB1 select the type of loopback to be performed. The NIC supports three modes of loopback: internal loopback through the DP8390 controller only (*Figure 1*), external loopback through the DP8391 encoder/decoder (*Figure 2*), and external loopback through the DP8392 transceiver (*Figure 3*).

### Loopback Operation in the NIC

To initiate a loopback test, a packet must first be assembled and transferred into the NIC buffer memory. Next, the Transmit Page Start Register, Transmit Byte Count Registers, and Transmit Configuration Register must be programmed. (When loopback mode is selected in the Transmit Configuration Register, the FIFO is split into two halves, one used for transmission and the other for reception.) Finally,

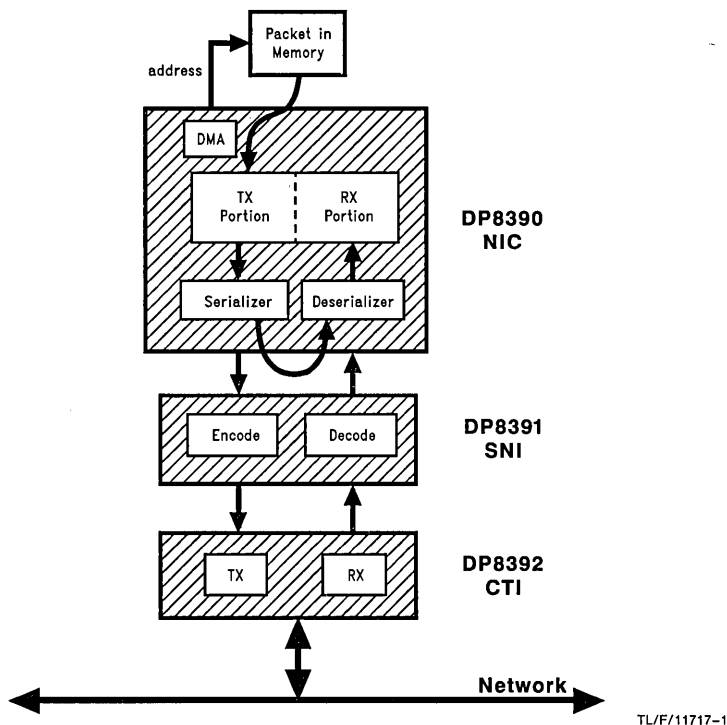


FIGURE 1. Loopback Mode 1: Through the Controller



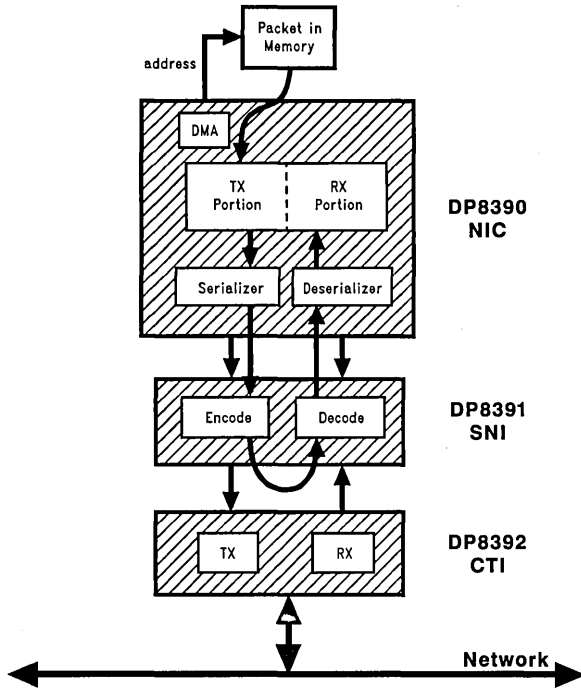


FIGURE 2. Loopback Mode 2: Through the SNI

TL/F/11717-2

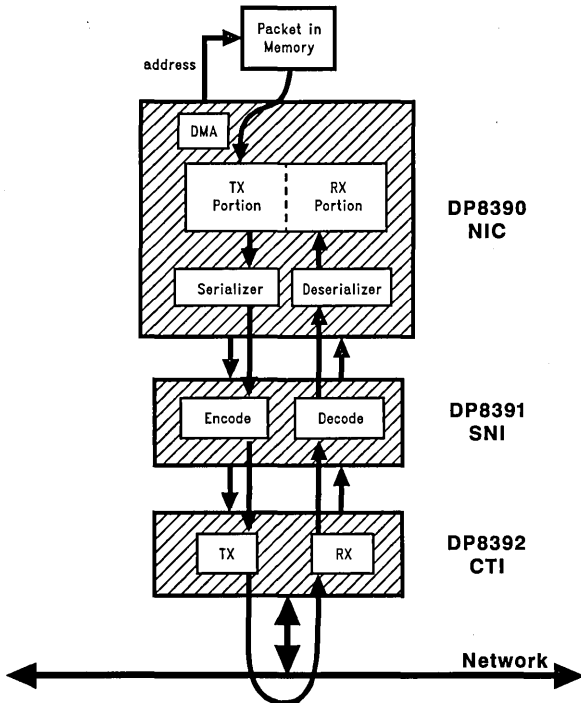


FIGURE 3. Loopback Mode 3: To the Coax

TL/F/11717-3

the transmit command is issued to the Command Register, causing the following operations to occur:

#### Transmitter Actions

1. Data is transferred from memory by local DMA until the FIFO is filled. For each transfer, the Transmit Byte Count Registers (TBCR0 and TBCR1) are decremented. (Subsequent burst transfers are initiated when the number of bytes in the FIFO drops below the programmed threshold.)
2. The NIC generates 56 bits of preamble followed by an 8-bit synch pattern.
3. Data is transferred from the FIFO to the serializer.
4. If the Inhibit CRC bit is set in the Transmit Configuration Register, no CRC is calculated by the NIC, and the last byte transmitted is the last byte from the FIFO (last byte of the packet). This allows a software CRC to be appended. If the Inhibit CRC bit is not set, the NIC calculates and appends four bytes of CRC to the end of the packet.
5. At the end of transmission, the Packet Transmitted bit is set in the Interrupt Status Register.

#### Receiver Actions

1. Wait for synch (Start of Frame Delimiter), all preamble bits are ignored.
2. Store packet in the FIFO, increment receive byte count for each incoming byte.
3. If the Inhibit CRC bit is set in the Transmit Configuration Register, the receiver checks the incoming packet for CRC errors. If the Inhibit CRC bit is not set in the Transmit Configuration Register, the receiver does not check for CRC errors; the CRC error bit is set in the Receive Status Register (for address matching packets).
4. At the end of receive, the receive byte count is written into the FIFO and the Receive Status Register is updated. The Packet Received Intact bit is typically set in the Receive Status Register even if the address does not match. If CRC errors are forced, the packet must match the address filters in order for the CRC error bit in the Receive Status Register to be set.

#### Restrictions Using Loopback

Since the NIC is a half-duplex device, several compromises were required for the implementation of loopback diagnostics. Special attention should be paid to the restrictions placed on the use of loopback diagnostics.

1. The FIFO is split into two halves to allow some buffering of incoming data. The NIC transmits through one half of the FIFO and receives through the second half. Only the last five bytes of a packet can be examined in the FIFO; the DMA does not store the loopback packet in memory. Thus loopback can be considered a modified form of transmission.
2. Splitting of the FIFO has some bus latency implications. The FIFO depth is halved, thus reducing the amount of allowed bus latency. The Loopback Select bit (D3) in the Data Configuration Register should be set to allow all local DMA transfers to continue until the FIFO is filled. In cases where the latency constraints cannot be accommodated, small 7 byte packets can be transmitted. In addition, the FIFO must only be read (by successfully reading port 06h) when in loopback mode; reading the FIFO in other modes will result in the NIC failing to issue the ACK signal properly.
3. The CRC logic is shared by the receiver and the transmitter; thus the NIC cannot generate and check the CRC simultaneously. That is, if the Inhibit CRC bit is not set in the Transmit Configuration Register, the NIC generates and appends the CRC, and software must be used to verify the CRC. On the other hand, if the Inhibit CRC bit is set in the Transmit Configuration Register, the NIC will verify a software generated CRC.
4. Address recognition logic must be checked indirectly through a small series of tests (see Group III Loopback Tests: Address Recognition for further explanation).
5. Between consecutive transmissions in loopback mode, the Transmit Configuration Register must first be set to 00h and the Command Register reset to 21h (followed by a wait state of at least 1.5 ms for the NIC to reset). The desired loopback mode may then be programmed into the Transmit Configuration Register. This step guarantees alignment of the FIFO pointers when data is read from the FIFO.
6. Loopback only operates with byte wide transfers, thus special considerations must be made with word wide transfers. Since the FIFO is split, only half of each word is transferred into the transmit portion of the FIFO. The Byte Order Select bit in the Data Configuration Register can be used to select which half of the word is written into the FIFO (see Figure 4).

**Note:** Although a word is transferred to the NIC, only a byte is transmitted in the loopback packet. To properly transfer all the bytes in the loopback packet, the byte count must be 2 times the actual number of bytes assembled in the loopback packet.

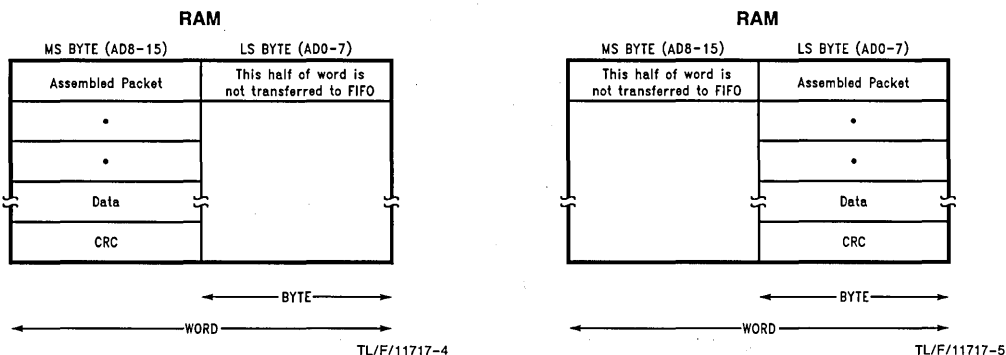


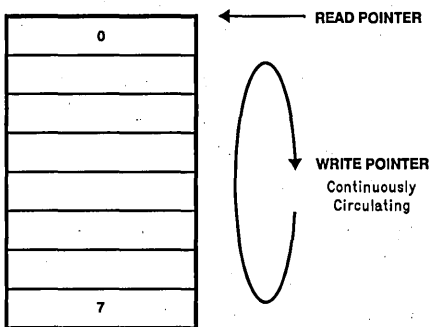
FIGURE 4. Packet Assembly for Loopback Work Wide Transfers

7. During heavily loaded network conditions, external loopback through the SNI and CTI could fail due to interference from the network.

**Alignment of Data in the FIFO**

During loopback, eight bytes of the FIFO are used for transmission and eight bytes are used for reception. Reception of the packet begins at location zero, and after the pointer reaches the last location in the receive portion of the FIFO, the pointer wraps back to location zero, overwriting the previously received data (see *Figure 5*). The pointer continues to circulate through the FIFO until the last byte is received. The NIC then appends the lower receive byte count and two copies of the upper receive byte count into the next three locations in the FIFO. Thus, only the last five bytes of the received packet may be retrieved.

**Note:** Although the size limit of a loopback packet is 64 bytes, the byte counter rolls over at 2048 bytes.



TL/F/11717-6

**FIGURE 5. Continuously Circulating FIFO Write Pointer During Loopback**

To achieve the packet alignment shown in *Figure 6*, the packet length should be  $(N * 8) + 5$  bytes (i.e., 13, 21, etc). If the CRC is appended, the second through fifth byte will be the CRC appended by the NIC. This allows the CRC to be extracted from the NIC and compared to a previously calculated value for verification.

FIFO Location	FIFO Contents	
0	Byte $(N * 8) + 1$	First Byte Read
1	Byte $(N * 8) + 2$ (CRC1)	Second Byte Read
2	Byte $(N * 8) + 3$ (CRC2)	•
3	Byte $(N * 8) + 4$ (CRC3)	•
4	Byte $(N * 8) + 5$ (CRC4)	•
5	Lower Byte Count	•
6	Upper Byte Count	•
7	Upper Byte Count	Last Byte Read

**FIGURE 6. Alignment of Packet in FIFO Following Loopback**

**Loopback Tests**

Three types of loopback tests may be performed to verify the data path through the DP8390 chip set. The tests are as follows:

1. Group I tests verify the CRC generation capability of the NIC. In this case, the NIC generates and appends a CRC to the loopback packet, and software is used to verify a matching CRC.
2. Group II tests verify the CRC recognition capability of the NIC. Here, the NIC verifies a software generated CRC.
3. Group III tests verify the address recognition logic of the NIC.

The loopback tests which follow were performed on the DP839EB. During each of the loopback tests, the Data Configuration Register was programmed to 40h.

**GROUP I LOOPBACK TESTS: CRC GENERATION**

The basic steps necessary to perform the Group I loopback tests (in which the CRC is appended by the NIC) are as follows:

1. Set Command Register to 21h (page 0).
2. Initialize Data Configuration Register to 40h.
3. Initialize Receive Configuration Register to 1Fh (promiscuous mode).
4. Initialize Transmit Byte Count Registers and Transmit Page Start Register.
5. Set Command Register to 22h (start mode).
6. Create loopback packet and transfer into NIC buffer memory.
7. Transmit dummy packet to check for unterminated or unconnected cable:
  - a. Set Transmit Configuration Register to 00h (normal operation).
  - b. Write FFh to Interrupt Status Register to reset.
  - c. Set Command Register to 26h (transmit). Note that the Command Register must first be in start mode (22h) before transmitting (26h).
  - d. Loop until the Packet Transmitted bit is set in the Interrupt Status Register. If the timeout loop completes and this bit is not set, the transmit has timed out, and the cable may not be connected.
  - e. Check Interrupt Status Register for 08h (transmit Error). If the Transmit Error bit is set, excessive collisions have occurred, and the cable may not be terminated.
8. Start loopback mode 1 test (TCR = 02h):
  - a. Reset Transmit Configuration Register to 00h.
  - b. Reset Command Register to 21h. If the NIC is currently receiving a packet, it will wait for the reception of the current packet to complete before it will reset. Thus, a wait state of at least 1.5 ms is necessary to insure that the NIC will completely reset.
  - c. Program the Transmit Configuration Register to the appropriate loopback mode.
  - d. Write FFh to Interrupt Status Register to reset.
  - e. Set Command Register to 22h (start mode).

- f. Set Command Register to 26h (transmit).
  - g. Wait for transmit to complete (Command Register = 22h).
  - h. Check Interrupt Status Register for 06h (good transmission).
  - i. Read FIFO and compare CRC with previously calculated CRC.
9. Start loopback mode 2 test (TCR = 04h): See Step 8.
  10. Transmit a dummy packet to change the contents of the FIFO. If this step is not taken before external loopback through the CTI and the AUI cable is not connected, the NIC does not receive anything into its FIFO. Thus the contents of the FIFO are not changed, and the loopback test reads a good CRC. See Step 7.
  11. Start loopback mode 3 test (TCR = 06h): See Step 8.
  12. If mode 3 loopback fails, transmission may have been aborted due to excessive collisions (check the Transmit Status Register). In this case, network traffic has interfered and the CTI may still be operational.

### GROUP I RESULTS

The following examples show what results can be expected from a properly operating NIC during Group I loopback operations. The restrictions and results of each loopback mode are listed for reference.

#### Internal Loopback through the NIC

Path	TCR	RCR	TSR	RSR	ISR
NIC Internal (Mode 1)	02H	1FH	51H	02H	06H

**Note 1:** Before transmission of the loopback packet, Carrier Sense and Collision inputs are monitored (as required by CSMA/CD protocol). Once the NIC gains access to the network for transmission, the Carrier Sense and Collision Detect inputs are ignored. Thus, the Carrier Sense Lost and CD Heartbeat bits are always set in the Transmit Status Register.

**Note 2:** CRC errors are always indicated by the receiver if the CRC is appended by the transmitter.

**Note 3:** Only the Packet Transmitted and Receive Error bits in the Interrupt Status Register are set; the Packet Received bit is set only if status is written to memory. In loopback this action does not occur, and the Packet Received bit remains 0 for all loopback modes.

#### External Loopback through the SNI

Path	TCR	RCR	TSR	RSR	ISR
NIC External (Mode 2)	04H	1FH	41H	02H	06H

**Note 1:** CD Heartbeat is set in the Transmit Status Register; Carrier Sense Lost is not set since it is generated by the external encoder/decoder.

#### External Loopback through the CTI

Path	TCR	RCR	TSR	RSR	ISR
NIC External (Mode 3)	06H	1FH	01H	02H	06H

**Note 1:** CD Heartbeat and Carrier Sense Lost should not be set. The Transmit Status Register could, however, also contain 01, 03, 07, and a variety of other values depending on whether collisions were encountered or the packet was deferred.

**Note 2:** The Interrupt Status Register will contain 08 if the packet is not transmittable.

During external loopback the NIC is now exposed to network traffic. It is therefore possible for the contents of both the receive portion of the FIFO and the Receive Status Register to be corrupted by any other packet on the network. Thus, in a live network, the contents of the FIFO and Receive Status Register should not be depended on. The NIC will still abide by the standard CSMA/CD protocol in external loopback mode (the network will not be disturbed by the loopback packet).

### GROUP II LOOPBACK TESTS: CRC RECOGNITION

The basic steps necessary to perform the Group II loopback tests (in which a software CRC is appended to the packet) are similar to those outlined previously for the Group I tests, with the following exceptions:

1. The loopback packet created must have a software appended CRC.
2. When programming the Transmit Configuration Register to the desired loopback mode, the Inhibit CRC bit must be set.
3. After the loopback packet has been transmitted, check the Interrupt Status Register and/or the Receive Status Register for CRC errors. If a CRC error has occurred, the loopback test has failed.

### GROUP II RESULTS

The following examples show what results can be expected from a properly operating NIC during Group II loopback operations. The restrictions and results of each loopback mode are listed for reference.

#### Internal Loopback through the NIC

Path	TCR	RCR	TSR	RSR	ISR
NIC Internal (Mode 1)	03H	1FH	51H	01H	02H

**Note 1:** Before transmission of the loopback packet, Carrier Sense and Collision inputs are monitored (as required by CSMA/CD protocol). Once the NIC gains access to the network for transmission, the Carrier Sense and Collision Detect inputs are ignored. Thus, the Carrier Sense Lost and CD Heartbeat bits are always set in the Transmit Status Register.

**Note 2:** Only the Packet Transmitted bit in the Interrupt Status Register is set. The packet received bit is set only if status is written to memory. In loopback this action does not occur, and the Packet Received bit remains 0 for all loopback modes.

#### External Loopback through the SNI

Path	TCR	RCR	TSR	RSR	ISR
NIC External (Mode 2)	05H	1FH	41H	01H	02H

**Note 1:** CD Heartbeat is set in the Transmit Status register; Carrier Sense Lost is not set since it is generated by the external encoder/decoder.

**External Loopback through the CTI**

Path	TCR	RCR	TSR	RSR	ISR
NIC External (Mode 3)	07H	1FH	01H	01H	02H

**Note 1:** CD Heartbeat and Carrier Sense Lost should not be set. The Transmit Status Register could, however, also contain 01, 03, 07, and a variety of other values depending on whether collisions were encountered or the packet was deferred.

**Note 2:** The Interrupt Status Register will contain 08 if the packet is not transmittable.

During external loopback the NIC is now exposed to network traffic. It is therefore possible for the contents of both the received portion of the FIFO and the Receive Status Register to be corrupted by any other packet on the network. Thus, in a live network, the contents of the FIFO and Receive Status Register should not be depended on. The NIC will still abide by the standard CSMA/CD protocol in external loopback mode (the network will not be disturbed by the loopback packet).

**GROUP III LOOPBACK TESTS:  
ADDRESS RECOGNITION**

The address recognition logic cannot be directly tested. However, the CRC Error and Frame Alignment Error bits in the Receive Status Register are set only if the address of the packet matches the address filters. Thus, if errors are expected to be set and they are not set, the packet has been rejected on the basis of an address mismatch.

**GROUP III RESULTS**

One method of testing the address recognition logic would be to transmit two loopback packets, one with a matching physical address, and one with a non-matching address. Both packets should have a CRC appended by the NIC. Expected results for each case follow.

**Internal Loopback through the NIC: Matching  
Physical Address**

Path	TCR	RCR	TSR	RSR	ISR
NIC Internal (Mode 1)	02H	00H	51H	02H	06H

**Note 1:** Before transmission of the loopback packet, Carrier Sense and Collision inputs are monitored (as required by CSMA/CD protocol). Once the NIC gains access to the network for transmission, the Carrier Sense and Collision Detect inputs are ignored. Thus, the Carrier Sense Lost and CD Heartbeat bits are always set in the Transmit Status Register.

**Note 2:** CRC errors should be seen in both the Receive Status Register and the Interrupt Status Register for an address matching packet.

**Note 3:** Only the Packet Transmitted and Receive Error bits in the Interrupt Status Register are set; the Packet Received bit is set only if status is written to memory. In loopback this action does not occur, and the Packet Received bit remains 0 for all loopback modes.

**Internal Loopback through the NIC: Non-Matching  
Physical Address**

Path	TCR	RCR	TSR	RSR	ISR
NIC Internal (Mode 1)	02H	00H	51H	01H	02H

**Note 1:** Before transmission of the loopback packet, Carrier Sense and Collision inputs are monitored (as required by CSMA/CD protocol). Once the NIC gains access to the network for transmission, the Carrier Sense and Collision Detect inputs are ignored. Thus, the Carrier Sense Lost and CD Heartbeat bits are always set in the Transmit Status Register.

**Note 2:** CRC errors should not be detected for a non-matching physical address.

**Note 3:** Only the Packet Transmitted bit in the Interrupt Status Register is set. The packet received bit is set only if status is written to memory. In loopback this action does not occur, and the Packet Received bit remains 0 for all loopback modes.

# Architectural Choices for Network Performance

National Semiconductor  
Application Note 873  
Larry Wakeman  
John Von Voros



## TABLE OF CONTENTS

### INTRODUCTION

#### NETWORK PERFORMANCE

##### Defining Network Performance

Where's the Bottleneck?

##### Measuring Performance

#### ARCHITECTURAL OVERVIEW

##### Hardware Interfaces

I/O Mapped Slave  
Shared RAM Slave  
Simple Bus Master  
Buffered Bus Master  
Intelligent Bus Master

##### Buffer Management Architectures

Transmit Requirements  
Receive Requirements  
Operating System Requirements  
Hardware Packet Buffering Schemes  
Simple DMA Buffering  
Buffer Ring DMA  
Linked List Packet Handling

#### SYSTEM APPLICATIONS

##### PC®/PC-AT® Client Adapter

##### PC (ISA Bus) Server Adapter

##### PC Motherboard Applications

##### PS/2® and EISA Server Adapter

PC System Bus (CPU Bus)  
PC System I/O Bus Design

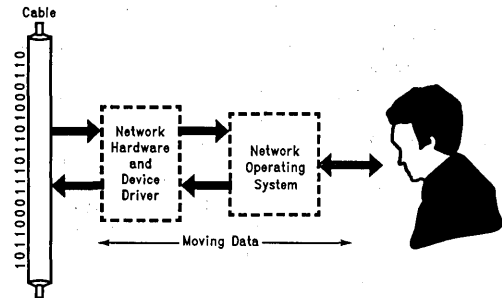
#### CONCLUSION

### INTRODUCTION

Recently the popularity of networking has grown, and as a result virtually any type of computer system and many peripherals are incorporating facilities to connect to a network. With the integration of the network function onto just a few IC's, the design of the interface circuitry to the network's physical interface is becoming simpler. However, the design of the interface can be implemented many different ways, with varying tradeoffs.

The basic design tradeoffs for interfacing a system to Ethernet are fairly simple:

1. **Performance.** Generally this is measured in terms of the amount of data transmitted and received in a given time period. The more data the better. As shown in *Figure 1*, the purpose of a Network interface is simply to: Move Data. The interface should be as fast and efficient as possible.



TL/F/11784-1

**FIGURE 1. The Success of a Network Interface Is It's Ability to Quickly, and Safely Move Data to/from the User**

2. **Low Cost.** Obviously the user would like to pay as little as possible for the network connection.
3. **Compatibility.** Both software and hardware compatibility to established industry standards is crucial. In the PC market, this means the ability to work with standard software (i.e., Novell's NetWare® and Microsoft's Windows for Workgroups®) and hardware. In the non-PC market, interoperability with other network components is the key to successful integration into an existing network.

Unfortunately, these simple goals can lead to choosing dramatically different designs for the Ethernet interface subsystem. The diversity of computer architectures (both hardware and software) requires a unique balance of all of these criteria.

This paper will concentrate on the application of Ethernet in PC type computer systems (i.e., Intel 286, 386, 486 CPUs). Both hardware and software issues will be addressed as they pertain to performance, cost, and compatibility. The considerations presented here are applicable to other computer systems as well.

### NETWORK PERFORMANCE

Obviously one of the major tradeoffs in developing a network interface solution is performance versus cost.

#### ***But: What is network performance?***

##### Defining Network Performance

Network performance is a measure of the ability of a particular network configuration to move data from one computer to another. Typically, this data movement occurs from a server to a workstation or client. Unfortunately, there is no standard method of measuring and benchmarking performance, due to the multitude of network and node configurations. We shall dissect the components of performance in an attempt to describe the roles that hardware and software play in a typical network.

When a user makes a request for information or data not resident on his own computer (like opening a memo in his word processor when the memo is located on a remote system), the network's pieces must all respond to this request. The user's perception of performance is determined by how long he must wait for the information to appear on his screen.

When an inquiry is made on the network for some information, a complex set of transactions occurs. The user's computer operating system tells the network protocol to send a message to the server asking for the information. The protocol software then instructs the driver to send the request to the hardware interface, which in turn sends data over the cable to the server. After the packet has been received, an acknowledgement is sent to the server indicating that a valid transfer was accomplished.

The reverse procedure occurs on the server end. When the hardware receives the request, the driver is instructed to pass it on to the network protocol. The computer operating system takes the request from the protocol and issues a response (the actual data) which is sent back through the network in a similar fashion.

Each of these software and hardware components manipulates the requests and responses to ensure proper delivery of the data to/from each destination. This process is shown in *Figure 2*. Each step requires time for the software and hardware to execute its piece of the job. The sum total of the time it takes for each operation to occur is the performance of the transaction.

#### WHERE'S THE BOTTLENECK?

There are many variables in the performance equation, and the network interface card is only one factor. Much like

Ethernet cards, the overall performance of a system can be divided into hardware and software issues. On the hardware side, the speed of a network is determined by the performance of the server and all of the attached workstations. The network operating system is a major contributor to the overall latency of a network.

The response time of a server or workstation is affected by CPU speed, bus bandwidth, network loading, and the speed and topology of the transmission media (such as coax, twisted-pair, optical fiber, etc.). The server should provide sufficient disk cache memory and a fast hard disk subsystem to minimize delays. Typically, throughput on a loaded network segment can be reduced to under 20% of maximum by random disk I/O requests. Improving individual workstation throughput has very little impact on the overall network since it only affects a small percentage of the total load. The expense of outfitting a high performance server can be amortized over the cost of the entire network since all users will benefit.

The network operating system (NOS) can also have a dramatic affect on Ethernet throughput. The two main functions of the NOS are to move large blocks of data around in RAM and manage the disk I/O subsystem. Excessive copying of data or poor file management will result in poor LAN performance.

Since each item in the request/response path contributes to overall performance, it is desirable to minimize delays through each section. From the Ethernet hardware developer's perspective, the efficiency of the NOS and the cable throughput are fixed. The only areas available for improvement of the network hardware performance are optimization of the driver and the bus interface design. It is important to recognize that the hardware and driver are a small (but important) part of the total performance equation. In most

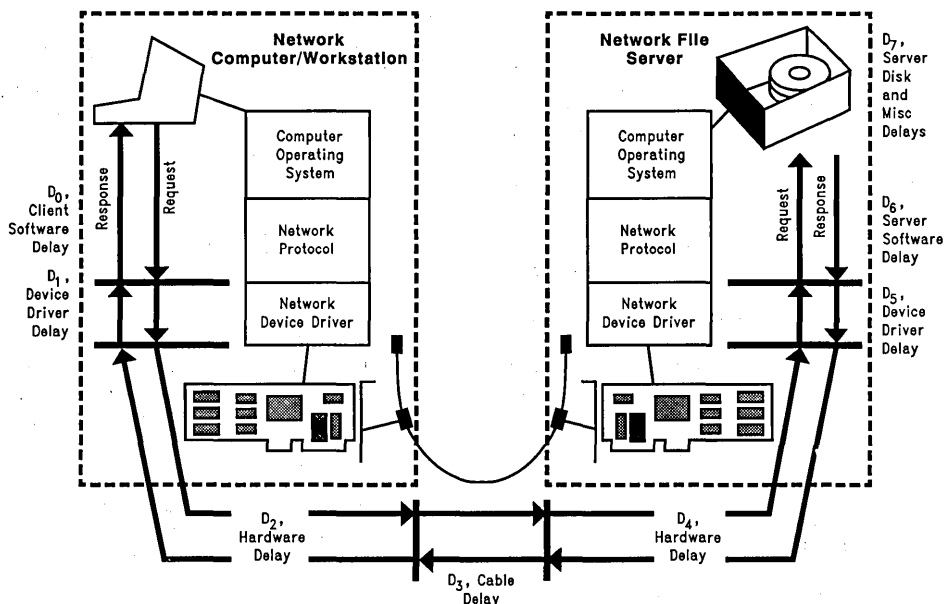


FIGURE 2. Representation of System Delays that Impact Performance

TL/F/11784-2

cases doubling or tripling performance of the network hardware will have a much smaller impact on total performance. (Whereas doubling or tripling the CPU performance could have a much greater affect.)

As can be seen in *Figure 2*, the total speed at which requests/responses can be handled is dictated by the serial path shown. For the analytically minded:

$$\text{Performance} = \frac{K_p}{\Sigma(\text{Request Delays})}$$

Basically, network performance is the reciprocal of the sum of the request delays for the measured transactions.  $K_p$  is a multiplier constant to convert to kbytes/sec.

A single request/acknowledge delay is the sum of the individual delays as shown in *Figure 2*.

$$\begin{aligned} \text{Request Delay} = & (D0 + D1) (2 K_{CPU1}) \\ & + (D5 + D6) (2 K_{CPU2}) \\ & + 2(D2 + D3 + D4) + D7 \end{aligned}$$

$D0$ ,  $D1$ ,  $D3$ ,  $D5$ ,  $D6$  are all software delays and thus are multiplied by the performance of the system processors, while  $D2$ ,  $D3$ ,  $D4$  are network hardware delays that are not affected by software performance (ideally).  $D7$  is the delay due to system hardware other than the network interface, such as a disk and controller.

### Measuring Performance

Generally, one would like to measure these delays and calculate the throughput of a particular network configuration. Most benchmarks can only evaluate the summation of these delays. This is done by measuring the actual data throughput, usually in kbytes/sec or in seconds for a particular task.

Now, of course every good marketer has his favorite benchmark, but the validity of a benchmark can be very deceiving. Most Ethernet vendors cite the Novell **PERFORMx** benchmark as a performance measure, this is valid but does not accurately model the request/response of a real network. Some testing labs have developed scripts that simulate user transactions in an attempt to create a more representative view of network speed, and these tend to be more valid. However, due to the ease of testing and general availability of the **PERFORM** program most comparisons utilize this program's measurements.

The perform benchmark tends to measure the throughput of the network interface, the protocol software, and the system CPU. The data being transferred is cached in the server's memory, so disk drive speed is not a factor. This means that the largest delay for a system is not taken into consideration.

Another important aspect of performance is the amount of time the server CPU is idle. The more CPU bandwidth that is available the greater the potential for the server to do other processing or to handle more information. This becomes important when the server is being used as a database engine or if multiple Ethernet cards are used concurrently. A server that is oversaturated will drop packets, and thus decrease performance because each of these packets will have to be retransmitted. Generally, CPU utilization indicates the potential for better performance rather than actual performance. In the NetWare world, the CPU utilization measure is made using the **MONITOR** program which is run on the server. A meaningful benchmark should contain both the throughput and the CPU utilization figures. For example,

a system with 100 kbyte/sec throughput with a 10% utilization (90% idle) should be better than a 100 kbytes/sec with a 90% utilization. When making comparisons between architectures, it is important to use the same equipment for each test since the benchmarks are also measuring the server's and workstations' performance. The network should have at least six workstations to ensure heavy Ethernet traffic.

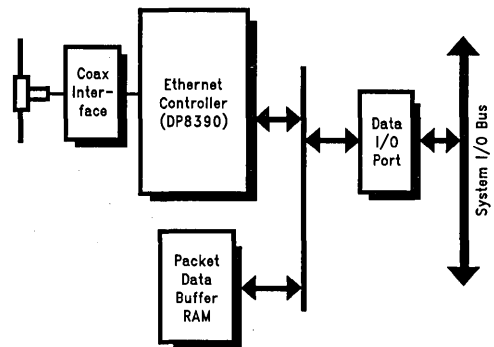
### ARCHITECTURAL OVERVIEW

The architecture of an Ethernet card encompasses a hardware interface to the system as well as a software interface, which is really the packet buffer management that the hardware implements with the device driver software. This section will cover the hardware aspects first since this is very much the systems designer's decision. Secondly a description of various software interfaces is described, although in most cases this is actually defined by the integrated controller chosen for a particular design.

### Hardware Interfaces

Before discussing actual applications, it is useful to categorize Ethernet interfaces. Once done, this can be applied to various applications to determine the best fit for each application. A summary can be found in Table 1. The interfaces from the Ethernet subsystem to the host's system bus can be divided into roughly 5 categories as follows:

1. **I/O Mapped Slave:** In this design, the adapter interfaces to the system via a limited number of I/O ports, usually 16 bytes-64 bytes. The interface has dedicated RAM to buffer network data, usually 8k to 64 kbytes. A simple block diagram is shown in *Figure 3*. National's DP8390 core controllers have on-chip logic to ease implementation of this interface.



TLF/11784-3

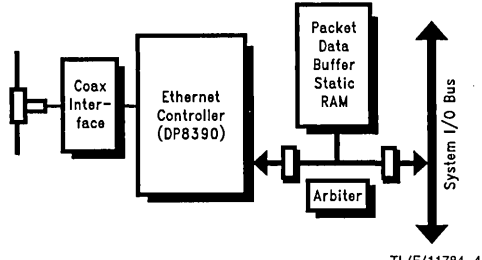
**FIGURE 3. Diagram Showing Major Blocks of the I/O Mapped Architecture**

**Advantages:** A very simple interface, tends to be low cost. Does not occupy large address space (important for PC's with many peripherals competing for common address allocations). Places little performance requirements on the system bus, and so is ideal for systems that have poor bus bandwidth or long bus latencies.

**Disadvantages:** May be slightly lower in performance. Requires dedicated buffer RAM which can add to cost of the interface.



**2. Shared RAM Slave:** This architecture utilizes a RAM that is dual ported (usually a static RAM with an arbiter rather than an integrated dual port RAM) to enable the network interface and the main system to communicate through a common "window" of memory, as shown in *Figure 4*. The DP8390 has request/acknowledge logic to simplify implementation of this interface.



**FIGURE 4. Basic Block Diagram of the Shared Buffer RAM Design**

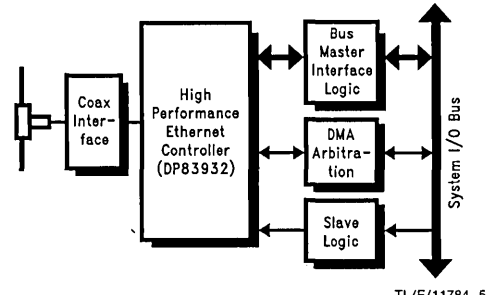
**Advantages:** A fairly simple interface that is slightly higher in performance than the I/O Slave since the data is directly accessible to the system via the buffer RAM.

**Disadvantages:** Tends to be more complex than the I/O interface, and thus more expensive. The additional cost is due to the logic required to dual port the buffer RAM which includes a couple of extra PALs, 4-6 extra octal buffers, and some added logic for software control. (However, as integrated or ASIC solutions become available this extra logic can be absorbed inexpensively making the solution cost equivalent to an I/O mapped design.) This architecture places slightly greater performance requirements on the system bus than the I/O Slave since the system is contending for the buffer RAM with the network interface.

The Shared RAM architecture is not the best choice when the host system has a limited addressing range and/or its memory cycles aren't significantly faster than its I/O cycles. In a PC-AT compatible application, the Shared RAM design typically has a 30% faster bus transfer speed, however when the effects of driver and Network Operating System (NOS) overhead are considered, the advantage of the Shared RAM design is reduced to 10% or less. In Micro Channel or EISA systems, the difference in I/O vs memory transfer rates is less, so the performance disparities would be reduced (assuming the network hardware does not present any other constraints).

**Note:** These relative numbers do not include disk access overhead, so the performance difference seen by a user will typically be lower.

**3. Simple Bus Master:** For the simple bus master interface, the network peripheral directly requests the system bus, and when granted, takes control of the bus and directly places packet data into system memory (*Figure 5*). The performance of this design is heavily dependent on the sophistication and speed of the DMA channel logic, and the bus itself. In the past, most Ethernet controllers did not have sufficient bus speed or buffer management to support this type of architecture, hence it's relatively new emergence has coincided with the introduction of high performance controllers such as the SONIC™ DP83932.



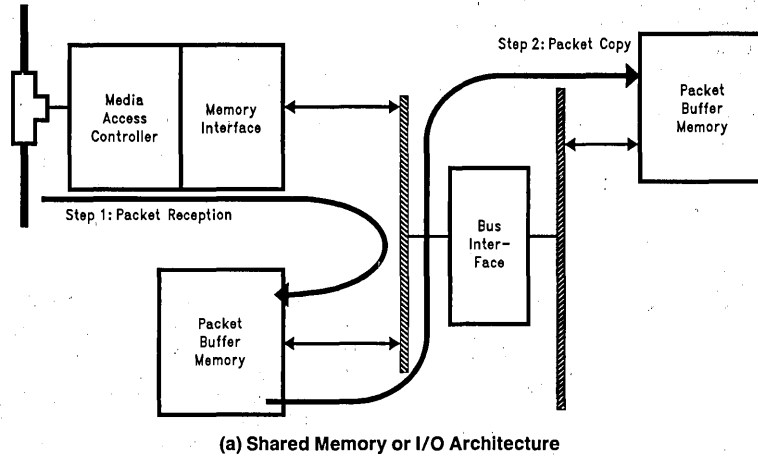
**FIGURE 5. Simple Bus Master Block Diagram**

In order to understand memory buffer structures, it is useful to compare the packet movement of a Shared Memory or I/O Interface (local memory) to that of the Bus Master design. In a local memory design (*Figure 6a*), the data is first buffered, then copied to the system memory. The Bus Master, on the other hand (*Figure 6b*), places the data directly into system memory. This latter approach is significantly faster because local memory designs require an extra read and write cycle to move the contents of the local buffer RAM into system memory. In theory, the Bus Master can eliminate additional data copies. Performance is reduced if the Bus Master cannot buffer directly to the NOS and a data copy has to be executed. The fact that local memory requires data buffering in two steps is not as significant to performance as the method of moving the data into system memory (i.e., DMA, I/O channel, etc.). Bus latency is one of the prime considerations when deciding to use a bus master approach. Newer generation controllers rely on a FIFO to buffer data until the bus becomes free for transfers. Depending on the computer, this delay can be longer than the maximum time allowed by the FIFO. When a FIFO overrun (or underrun) occurs, the packet must be retransmitted. In theory, the maximum bus latency tolerated by a controller can be calculated by the equation:

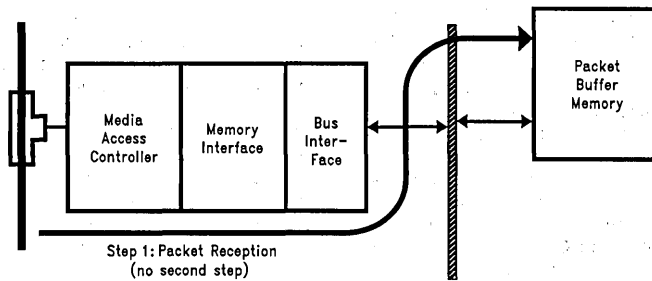
$$\text{Latency}_{\text{max}} = \frac{\text{Depth of FIFO (Bytes)}}{10 \text{ Mbits/S} \times 0.125 \text{ Byte/Bit}}$$

There are two classes of bus masters which for this document we will call a MAC (Media Access Control) Bus Master, and a Bus Master (for lack of better names). The distinction is that a MAC Bus Master becomes owner of the bus, but utilizes some form of system or external DMA controller to actually move the data. In other words the MAC Bus Master cannot generate addressing for the received or transmitted data. The Bus Master, on the other hand, utilizes a DMA controller that is built into the MAC. This DMA controller is capable of controlling data movement in a reasonably sophisticated way, and can place data into or take data from any desired location.

**Advantages.** Properly designed with enough intelligence in the packet buffering (i.e., not typically a MAC Bus Master), this implementation provides a very high performance data transfer throughput. If the DMA master is sophisticated enough, data can be placed directly into system memory, thus eliminating extraneous data copying by the driver software as is required by I/O mapped and Shared RAM designs.



(a) Shared Memory or I/O Architecture



(b) Bus Master Architecture

FIGURE 6. Comparison of Data Movement

**Disadvantages.** In order to achieve high performance, the DMA machine must be sufficiently sophisticated (not a MAC Bus Master). In many low performance bus systems, direct bus ownership is not supported. In other buses, such as ISA, the bus is not sophisticated enough to arbitrate between potential bus owners without tying up the CPU unnecessarily. In some high performance bus interfaces, the latency from bus request to bus grant can be very long and require on-card buffering of the data to avoid dropping packets.

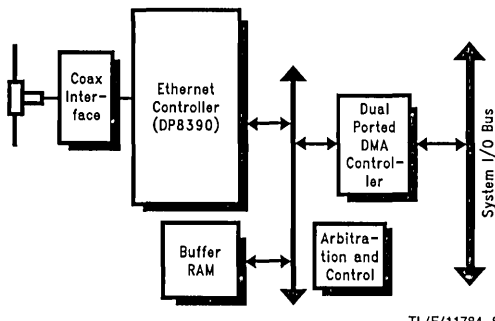
When comparing software driver performance, the efficiency of the driver plays a bigger role in Bus Master designs. This is because the Bus Master's hardware transfer is very efficient and the overhead of the driver is a bigger percentage of the data throughput. Typically in a PC (ISA bus) the bus master data transfer rate is 2-2.5 times that of an I/O based design. When software overheads are included, however, the Bus Master design typically achieves a performance increase over the I/O design. (Driver inefficiencies in reality can reduce this by about 5%.)

**Note:** These relative numbers do not include disk access overhead, so the performance difference seen by a user will typically be lower.

4. **Buffered Bus Master.** In this design the network packet data is DMA'd by a network controller through the on-card bus into a buffer RAM (Figure 7). The packet data is then transferred to the system by additional logic that DMA's the data across the system bus into main memory. The performance of this architecture is comparable to that of the standard bus master with a marginally higher use of CPU bandwidth.

**Advantages.** This architecture provides high performance close to the simple bus master design even in situations where there are extremely long bus latencies (i.e., EISA).

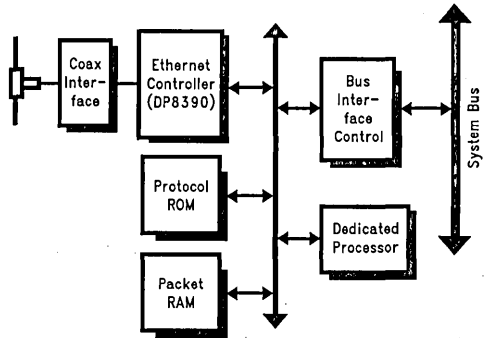
**Disadvantages.** When compared to the simple bus master interface, the cost of implementation is higher since this design requires additional buffer RAM and a complex system bus DMA channel in addition to the network interface's DMA channel. If the DMA interface is not sophisticated, the performance will be lower and the software driver will have to do additional processing.



TL/F/11784-8

FIGURE 7. Buffered Bus Master Block Diagram

5. **Intelligent Bus Master.** This design has a general purpose processor dedicated to the network interface for processing packet data (Figure 8). For low-end cards, the processor does not do protocol processing but only performs packet data manipulation and controls access between the system and the network interface. On high end designs, the dedicated network CPU does protocol processing which off-loads this task from the main system. The transfer of data to the system may be via an I/O, shared RAM, or bus master interface.



TL/F/11784-9

FIGURE 8. Intelligent Bus Master Block Diagram

**Advantages.** When using a processor with sufficient performance, this solution offers the highest performance of any of the solutions. This design also allows for the highest bus latencies, since on board RAM can store many packets. This architecture can off-load the server's CPU from processing the low level protocol tasks and can thus achieve very low server utilization relative to other technologies.

**Disadvantages.** Very costly in terms of hardware and component count. In order to achieve significant packet throughput advantages, the dedicated processor must be able to process packets at least as fast as the host CPU. In many cases, the low end cards are less efficient than simple bus master cards in terms of packet throughput.

In most practical examples, medium performance 16-bit processors are chosen and this choice tends to offer lower packet throughput than any of the other architectures. Typical CPU loading is roughly half that of a non-intelligent bus master.

**BUFFER MANAGEMENT ARCHITECTURES**

Just as the performance of a particular hardware implementation depends on how fast data is transferred to the system, the packet buffer management scheme helps determine how fast data can be transferred from the hardware to the Network Operating System. A great hardware design can be foiled by a poor software interface.

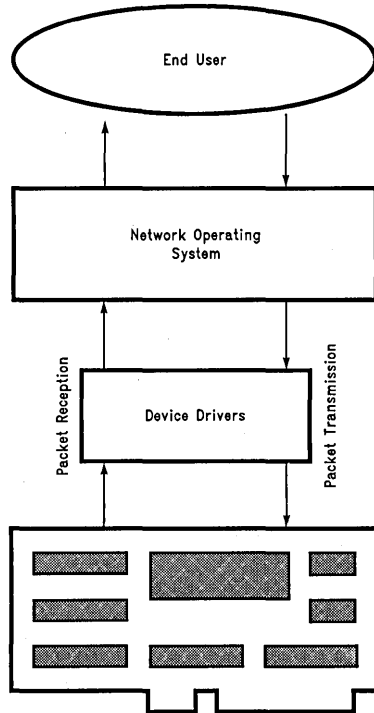
High performance software designs will reduce software complexity and directly provide the data to the NOS in a way that the NOS expects. There are cost/performance trade-offs in this interface as well, and so there are various buffering methods.

Before discussing the types of buffering that hardware may choose to implement, it is first useful to look at what operating systems expect for packet data. The goal is to minimize the device driver overhead, so a look at what information and in what form the NOS expects it is important.

Figure 9 shows a representation of how the user sends and receives data to/from the NOS. For sending packets, the NOS breaks up the data into smaller pieces so that they can fit within an Ethernet frame. The driver then takes this data and presents it to the hardware. On reception, the hardware gives data to the driver which in turn passes this data to the NOS. The NOS translates the data (if necessary) to a form acceptable to the user's application.

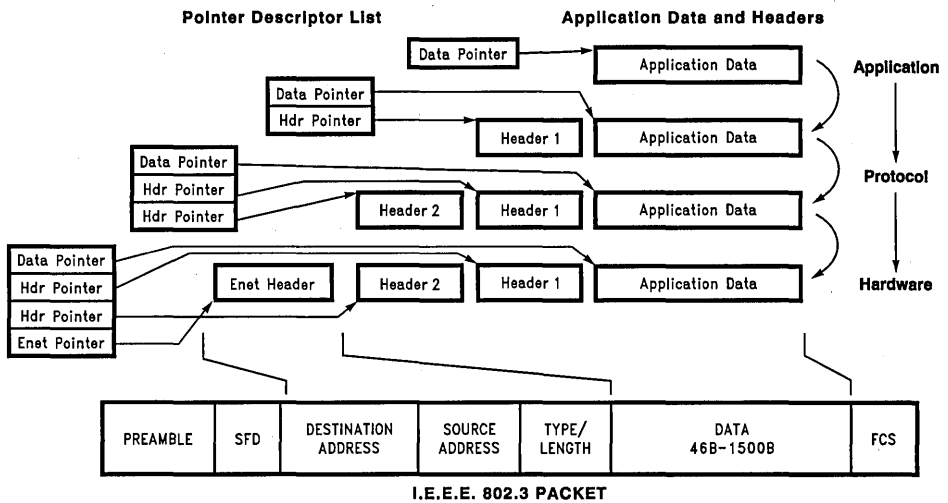
**Transmit Requirements**

The simpler part of the packet buffering scheme is the transmission of a packet. In this instance, the packet starts from the user's application, and the NOS prefixes network data, usually referred to as headers, to the application data as shown in Figure 10. These headers can contain protocol, routing, or application specific information. The most efficient method of "prefixing" is to create a pointer list which describes the data pieces' locations rather than copying all the data into a contiguous area. The driver receives this list from the NOS and then sends the information to the hardware. The network interface controller should be able to use this list with as little driver software manipulation as possible. The hardware and its associated driver must also be able to queue multiple requests since the network cable can only send one packet at a time.



TL/F/11784-10

FIGURE 9. User-Cable Data Movement through NOS to Driver to Hardware



TL/F/11784-11

FIGURE 10. Transmit Packet Creation

## Receive Requirements

One might assume that the receive packet handling should be very similar to the transmit, but there are a number of differences. First, unlike a transmit packet, the NOS, driver, and hardware have no idea when a packet may arrive, what kind of packet (i.e., IPX, TCP/IP, DECnet, etc.) it is going to receive, and how many packets may be received in a given time. While a packet to be transmitted is statically resident in memory until it is operated on, a packet being received must have sufficient memory allocated to it prior to reception. The amount of memory set aside must be equal to the maximum packet size since there is no way to predetermine a packet's length.

These unknowns require a different type of buffer management. Since a packet could be received at any moment, the driver or the hardware must allocate memory before the packet arrives. The system should provide enough memory to handle several packets at a time in case the packets are received faster than they can be processed. This memory allocation is accomplished in hardware by most architectures. A dedicated packet buffer RAM on the network card allows sufficient space to receive multiple packets (this capability is the reason that dedicated hardware RAM is used). All but the simple bus master architecture have a dedicated packet buffer.

In the simple bus master, the driver must allocate a dynamic pool of system memory, and so the structure of the receive portion of the driver depends more on the memory allocation scheme of the NOS than the network side. Thus for simple bus masters to be effective, they should implement a memory allocation/management scheme similar to that of the NOS to simplify data manipulation.

Another consideration is that ultimately the packet will be given to the host's operating system, so the hardware/driver should present the data in a compatible format. The NOS will fragment the packet to remove all of the headers and give the data to the receiving application in the form it accepts.

As can be seen in *Figure 11 a-c*, there are several possible schemes for dealing with received packets. In cases where only one packet type is being received, the hardware/software may be able to fragment the packet into its multiple headers as it is received and to place each header into a separate area for manipulation (called protocol fragmentation buffering). In some cases, this scheme is efficient since data copying can be eliminated (in theory). The down side for the NOS is that each layer must keep track of several pointers. When multiple protocols and packet types are involved, this type of scattering is difficult because the hardware will have to receive enough of the packet to determine

its type, and then either the hardware or software must find appropriate memory to place the fragments. This effort is required because different packet types have different header lengths and contents.

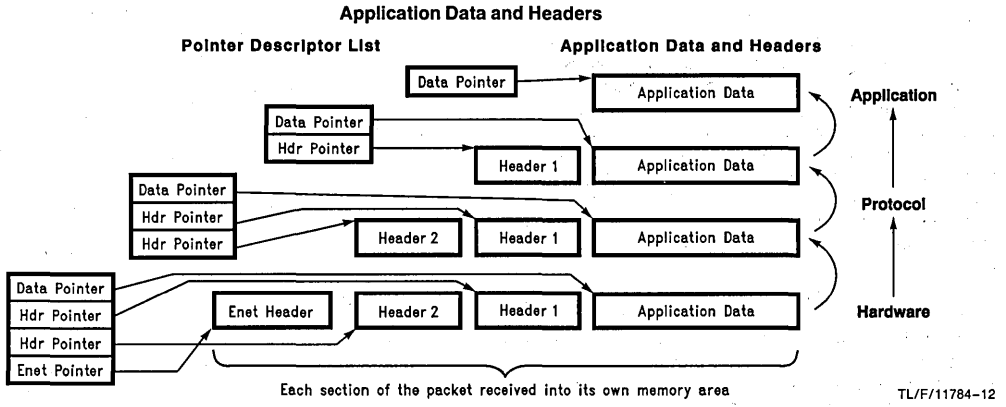
Another possible reception method is to fragment the packet into small buffers, usually 256 or 512 bytes. This will result in a large Ethernet packet being chopped into 3 pieces (*Figure 11b*). There are several advantages to this technique:

1. Operating system memory management uses these block sizes, so memory allocation is simplified by not having to allocate large contiguous memory blocks.
2. Most Ethernet packets are relatively small, usually <256 bytes. This type of memory scheme is more memory efficient than if a packet were contiguously buffered to a single area. This is because each memory area must be able to accommodate a full Ethernet packet 1518 bytes, and if a small 100 byte packet is received then the rest of the packet buffer will contain unused memory. (However these days operating systems with huge multi-megabytes of memory are common, and a few extra kbytes of packet buffers is typically not a big problem.)
3. In some schemes, the beginning of a new packet may be buffered directly at the end of the previous packet which causes additional fragmentation and hence more pointers (but is more memory efficient).

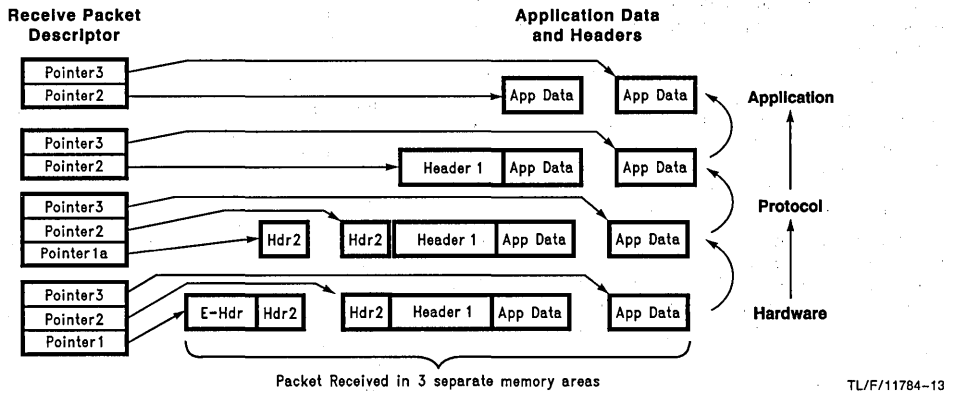
The problem with this second scheme is performance. When a packet is scattered as shown in *Figure 11b*, multiple pointers are required to keep track of the packet, this means that there is more software overhead associated with maintaining the pointers, and it is possible that some fields within the packet may be split between two buffers. Fortunately, this split is not likely to occur in the packet headers if >64 byte buffers are used, but the application data will be split, and may need to be copied to a contiguous buffer by the NOS prior to handing it off to the receiving application.

The problems with the two associated schemes can be overcome by having the hardware buffer the packet into a single contiguous memory area (*Figure 11c*). This allows the protocol software to have only one pointer to describe a packet. When a header is processed, the old pointer is thrown out, and the next header's pointer is easily created. The packet data arrives at the application in contiguous form. Multiple pointers to packet headers can easily be created if the software requires it. In multiple packet type applications, it is easier/cheaper for the software (as opposed to hardware) to determine the type of protocol/packet type for the received packet and manipulate the data based on this determination.

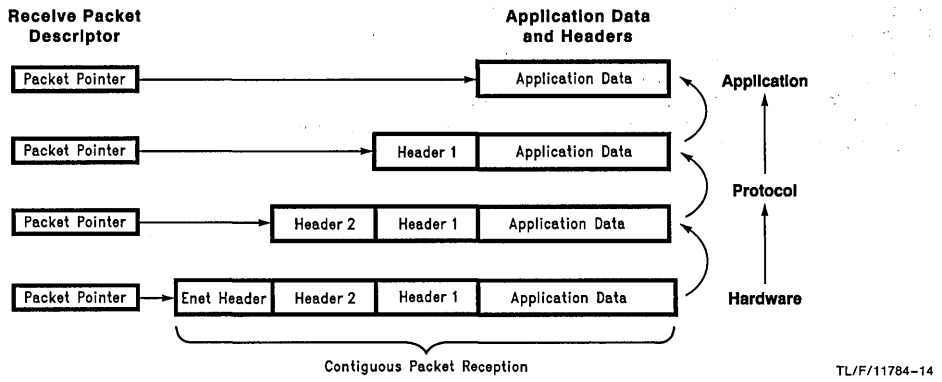
POINTER DESCRIPTOR LIST



(a) Receive Packet Scattered by Section



(b) Received Packet Scattered Based on Small Allocated Memory Blocks



(c) Received Packet Buffered to a Contiguous Memory Area

FIGURE 11

### Operating System Requirements

As has been described, the optimal buffering scheme for packets depends, in part, on the way the NOS interacts with the device driver and the hardware. At the device driver level, the NOS defines a programming interface for the exchange of packets between the hardware/driver and the NOS. The type of interface helps to determine the desired buffering scheme.

**Note:** Hardware architecture also affects this choice.

For non-embedded applications which use standard operating systems, the most prevalent transmit schemes provide the driver with a list of locations for the various portions of the packet. The driver/hardware then assembles the packet to prepare for transmission. Several schemes are used for receiving packets, but contiguous packet reception is the most popular. Packet scattering based on small memory block allocation is also quite common.

Most operating systems require a certain byte alignment on received packets to conform to their memory management schemes. For example, 32 bits OS's usually require double word alignment, while PC DOS (an 8-bit OS), most often demands byte alignment. In general, the transmit alignment is usually byte oriented because the header fragments generated may be an odd number of bytes.

### Hardware Packet Buffering Schemes

Keeping in mind the general characteristics outlined above, several hardware packet buffering techniques can be compared. The NOS requirements do not change based on the hardware architecture or the buffering scheme chosen, so when the hardware does not provide optimal algorithms, the device driver software is required to complete the job.

When the hardware packet buffering scheme minimizes bottlenecks (particularly software overhead), the theoretical performance of the driver/hardware set will increase. This section compares major buffering schemes and how they map into the NOS operations, hopefully revealing an indication of the better architectures.

The several schemes can be categorized as follows:

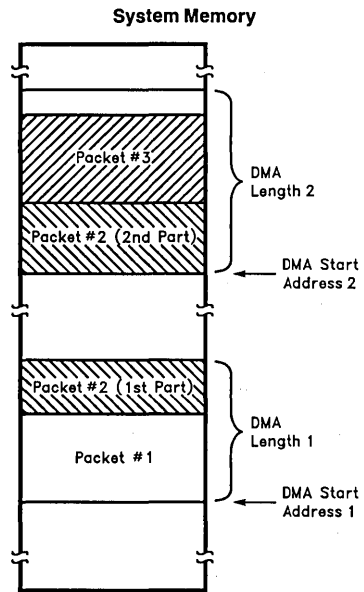
1. Simple DMA: Utilizing a simple start address and length, the system DMA performs all memory transfers.
2. Buffer Ring: A block of memory is setup as a recirculating ring where data at the top of the memory block automatically wraps to the bottom, and pointers track the used/open memory space.
3. Linked Lists: A number of descriptor structures that describe blocks of memory. Each block can contain either a part of a packet, a single packet, or multiple packets.
4. Protocol Translation: This scheme must be implemented on an intelligent card since the on board CPU performs driver tasks as well as protocol processing. The designer can customize the hardware using any combination of the three previous buffering schemes if the native CPU is not needed for protocol translation. This buffering scheme, due to it's unique programmability, will not be discussed separately.

### SIMPLE DMA BUFFERING

Typically this buffering architecture is used when the Ethernet Hardware is a simple MAC (Media Access Controller) Bus Master card that uses the system's DMA to provide a low cost solution. It is possible that the System DMA can be used in conjunction with a hardware scheme that includes a dedicated buffer RAM (like Shared RAM).

Reception tends to be a problem for the simple MAC bus master, so this is discussed first. Incoming packets are buffered in a small FIFO and a request is made for the system DMA controller to transfer the data. Simple bus master cards that do not use local memory must have access to the system bus before the FIFO fills, or the packet will be dropped. In addition, the host CPU must be able to allocate new blocks of memory as they are needed, which can be significant in terms of software overhead. For ISA based PC's, the DMA transfer rate is between 1.0-2.0 Mbytes/second which is not sufficient to keep up with the Ethernet data rate.

The transmit operation requires the DMA controller to transfer data from the host's memory to the controller's FIFOs (Figure 12). If the FIFOs are not large enough to buffer the maximum packet size, care must be taken to avoid a FIFO underrun because partial packets will be transmitted on the network. Once again, the host's CPU is responsible for all memory management.



TL/F/11784-15  
**FIGURE 12. System Memory Packet Reception**

### Buffer Ring DMA

For this architecture, the controller's memory manager utilizes a ring structure comprised of a series of contiguous fixed length buffers in a local RAM (Figure 13). The ring is typically divided into a transmit and receive section by the driver software. During reception, incoming data is buffered to the receive portion of the ring and then transferred to the system by the local DMA channel. The memory manager is responsible for three basic functions during reception: linking receive buffers for long packets, recovery of buffers when a packet is rejected, and recirculation of memory blocks that have been read by the host.

When transmitting data, the software driver must first assemble the packet in the transmit portion of the ring using DMA transfers. This information must include the destination address, the source address, the type/length of the packet, and the data itself. When transmission begins, the controller's local DMA channel transfers the data out of the ring and into the controller's FIFO. The controller sends out the data and appends a CRC field. The block of buffer memory used by the packet is then returned for reuse.

### Linked List Packet Handling

In a linked list structure, packets that are received or transmitted are placed in buffers that are linked by lists of pointers. The advantage to this approach, as mentioned earlier, is that it should eliminate unnecessary packet copying. The

software driver pre-allocates memory for receiving data and stores a list of pointers to available buffer pages in a Receive Resource Area (Figure 14). Another list of pointers (Resource Descriptor Area) is created when packets are received. Each pointer in this list corresponds to the starting address in memory of the received packet. Multiple packets can be stored in the same buffer area as long as their total length does not exceed the buffer page size.

The transmit buffer management scheme uses two areas in memory for transmitting packets (Figure 15). The Transmit Descriptor Area contains several fields that describe the packet to be transmitted and a pointer to the descriptor of the next packet to be transmitted. Quite often, operating systems store packet header information in one portion of memory and application data in another. Each of these fields is called a fragment.

The typical Ethernet packet contains multiple fragments, so the linked list buffering scheme provides pointers to each piece as well as a count of how many fragments are in the packet. In contrast, the buffer ring architecture would have required the driver to copy all of the fragments and the applications data into a contiguous area of local RAM prior to transmission. The buffer ring is a simple lower performance packet buffering scheme. The linked list structure adds some complexity, but will increase performance when properly tailored to the network operating system.

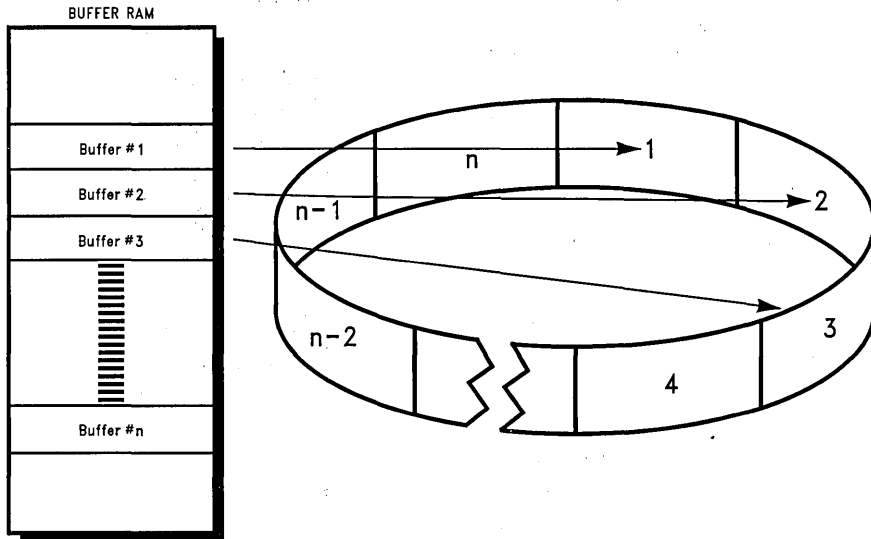


FIGURE 13. Buffer Ring DMA Structure

TL/F/11784-16



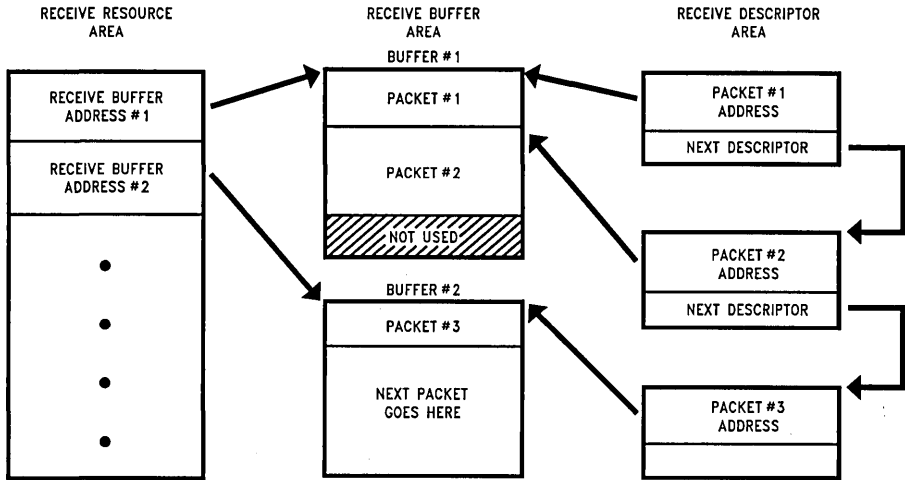


FIGURE 14. Linked List Packet Receive Buffer Structure

TL/F/11784-17

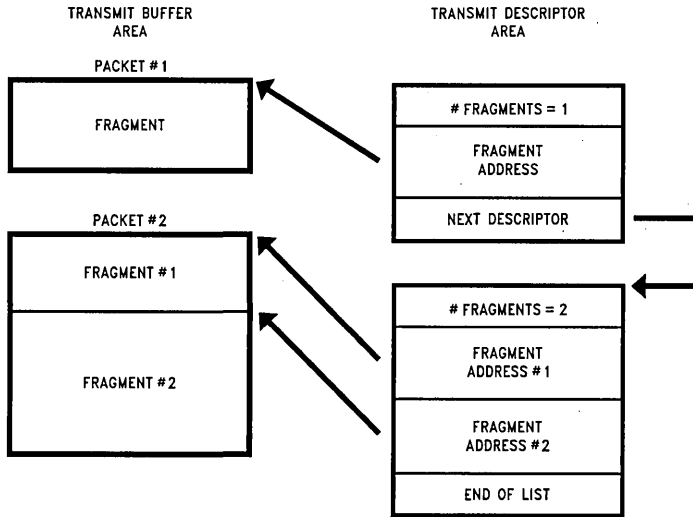


FIGURE 15. Linked List Packet Transmit Structure

TL/F/11784-18

## SYSTEM APPLICATIONS

For a certain application, many architectures may prove adequate, but the *best* solution may not be obvious. The following section will discuss the general design tradeoffs to each approach as they apply to personal computers and office peripherals.

### PC/PC-AT Client Adapter

While many companies promote performance, most simple interfaces prove to be sufficient, therefore cost and compatibility are of greater importance. The I/O mapped design is by far the most prevalent architecture for PC client cards due to good packet throughput performance and low cost. I/O mapped cards also tend to be the easiest to install because the I/O space of PC's tends to be less crowded than the memory space.

Shared memory cards must map their local packet RAM into the PC's address space between 640k and 1M. In the DOS environment, this space is crowded with BIOS ROMs, EGA/VGA video RAM, and disk controller hardware. Depending on the machine, configuration address contention can result. Also, the PC-AT bus timing for dual ported RAMs is tricky and varies somewhat between clones, thus making compatibility a more difficult issue.

For bus mastering cards, arbitration is not well implemented on many PC-AT compatibles and is not available in the PC-XT. These types of cards seem to have the most trouble interfacing to PC-AT clones because they are sensitive to system timing and the addition of other add-in cards. An improperly designed card can interfere with DRAM refresh and thus cause catastrophic failure. Earlier Ethernet controllers are not suitable for this approach because the bus cycles are excessively slow, thus limiting CPU performance.

**Note:** PS/2's with a microchannel bus provide very good arbitration, so bus masters are much more suitable for this environment.

The other architectures previously mentioned tend to be too expensive for the minor performance gains that would be achieved.

**Performance.** Assuming that the I/O port design is the reference, then the relative performance of the shared RAM design offers between a 2%–7% packet throughput improvement. Faster 386 based machines tend to minimize any advantages when the NOS is taken into consideration. The bus master could offer up to 10% speed improvement, but when using older Ethernet controllers this improvement is less. The major advantage to this approach can be lower CPU utilization.

### PC (ISA Bus) Server Adapter

Until very recently, servers have been mostly high performance ISA Bus PC-ATs. Due to multiple users, the traffic on the server is much higher than the client, and so bottlenecks in packet transfer will be more noticeable. Servers may also have to support multiple network cards to allow for the internal bridging of networks. The weakest link in these systems tends to be the relatively slow ISA bus.

For ISA bus servers, I/O mapped, shared RAM, and bus master designs all have their advantages and disadvantages. The best overall solution could be the I/O mapped design in spite of its slightly lower performance; it is compatible to industry standards, lower cost, and offers reasonable performance. Multiple cards can easily be employed since this interface does not put a burden on the bus, nor tie up needed RAM space.

The Dual-Ported RAM design offers slightly better performance, and won't tie up the bus, but does use precious memory space. This solution may prove less suitable if several cards are required in the Server.

Simple bus masters can provide the best performance, but since multiple cards can tie up the PC-AT's ISA bus and prevent CPU and refresh from getting sufficient access to the bus, this could present a problem when multiple cards are placed on the bus. No standard exists for supporting arbitration among multiple bus masters. Some initial implementations will only allow installation of one card due to slow bus cycles.

**Performance.** Assuming that the I/O port design is the reference, then the relative performance of the shared RAM design offers between a 2%–7% packet throughput improvement. The bus master could offer up to 10% speed improvement.

### PS/2 and EISA Server Adapter

The 32-Bit 386/486 PS/2 and EISA machines require the best performance, and cost tends to be a secondary issue. In addition, both these buses have intelligent bus arbitration schemes for bus ownership. The major difference between the two is that the arbitration scheme for EISA has the potential for having a relatively large bus latency, whereas the Micro Channel bus latency tends to be lower. This difference affects the bus mastering approaches taken.

The I/O mapped scheme is not optimal since cost is less of an issue and performance is more important. The Dual Port RAM scheme is a better choice as bus transfer speeds can be optimized by using fast RAMs, but the cost of the associated RAMs and logic is more (4-32kx8 SRAM are typically used for on-card buffering).

A simple bus master can be a very good choice if the desired bus latency is accommodated and an intelligent buffering scheme is implemented. This architecture can cost the same as a shared memory design, but provide faster packet throughput. Also multiple cards can easily be accommodated.

For the best performance, a well implemented intelligent board which does on board protocol processing is the best choice. However, the cost is prohibitive, and while overall server CPU usage can be minimized, typical implementations do not offer the best overall throughput.

### PC Motherboard Applications

The goals in designing Ethernet connectivity onto PC motherboards differs from those of PC add-in cards. First, due to severe competition and a network oriented focus, add-in board designers tend to concentrate on both the cost and performance of an implementation. The primary concern for PC motherboard designers is CPU performance and compatibility to existing standards. The purpose of including Ethernet is to provide added value and a simple inexpensive connection. Board space and power consumption tend to be more critical on motherboards.

Applications on the motherboard fall into two design categories:

1. An adapter card design folded down onto the main system board.
2. A system bus interface (or local bus) directly connected to the CPU.

The best approach depends on whether the Ethernet's design goal is primarily cost or performance driven. The folded-down design offers compatibility with well established standards and thus the lowest risk. The system interface architecture offers better performance at the expense of complex system design considerations.

#### PC System I/O Bus Design

The easiest approach to embedded Ethernet is to simply graft an existing PC adapter's design onto the motherboard or daughter card. This places the controller in a less performance critical area of the overall system design (the I/O bus) and allows a migration path from a solution that is known to work. Since backward compatibility is achieved, investments in software and experience are preserved. This design can be applied across an entire line of PC's with no modification to the hardware or software.

The most common implementation would be to "fold" an ISA bus 16-bit Ethernet card design onto the motherboard and thus provide a common interface for both ISA and EISA. The growing popularity for ISA based adapters has led semiconductor suppliers to provide very highly integrated solutions for this environment. Unfortunately the same integration level is not yet available for EISA based 32-bit designs.

The only disadvantage to "folding down", an adapter card solution is slightly lower performance. Since the throughput of Ethernet is usually "cable limited" this approach is suitable for clients and most servers.

#### PC System Bus (CPU Bus)

In this implementation, the Ethernet controller is tightly coupled to the system CPU bus (386 or 486). This is illustrated

by the top shaded block in *Figure 16*. In a PC, the highest performance bus is the CPU system bus. Ethernet controllers designed to operate in this environment can provide a relatively clean interface with a low parts count. The bus master architecture makes the most sense for this bus due to the simplicity of the interface.

The CPU system bus tends to be the most critical aspect of a PC's overall performance. Adding peripherals to this bus has generally been avoided because I/O functions can reduce bus efficiency. Embedded cache memories on some CPU's help to lessen this burden, but system performance will be affected. Another concern is that this bus was not designed to support the large fanout required for driving multiple I/O devices.

Interfacing to the CPU's bus presents many challenging design problems. The characteristics of this bus are determined by both the CPU and the memory controller. Changes in CPU type and frequency will cause the interface to vary for each PC model in a product line. The controller's operating frequency must be closely matched to that of the CPU to avoid timing problems. This can create problems for modular PC's that offer CPU upgradeability.

Table I summarizes the discussion on PC-Ethernet architectures. It should be noted that the ratings assume that each implementation is a good efficient design. For example, a simple bus master is an excellent architecture only for applications where it meets with the requirements of the bus; this may not always be true. Some qualitative performance references are given, but these should not be taken as valid for every case.

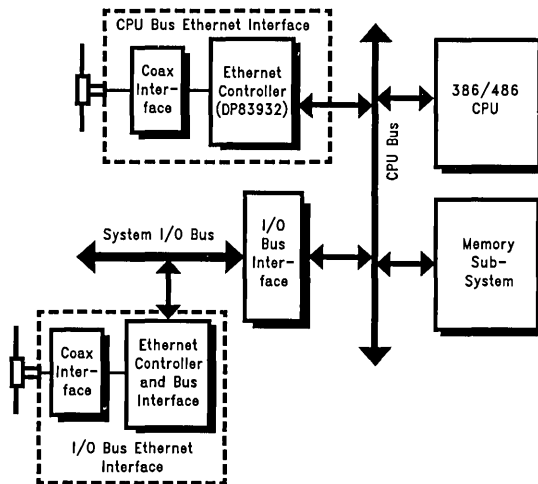


FIGURE 16. Two Choices for Ethernet on a PC Motherboard

TL/F/11784-19

TABLE I. Suitability of Architecture for PC Applications

Architecture (Note 1)	I/O Mapped	Dual Port RAM	Simple Bus Master	Buffered Bus Master	Intelligent
PC, PC AT (ISA), PS/2 (Client) (Note 2)	Excellent	Good	Fair	Poor	Poor
PC AT (ISA Bus) (Server) (Note 2)	Good	Good	Fair	Poor	Fair
PS/2 (Server) (Note 2)	Fair	Good	Excellent	Fair	Excellent
PC AT (EISA) (Client/Server)	Fair	Good	Excellent	Good	Good
PC Motherboard (System Bus) (Note 3)	Good	Good	Good	Poor	Poor

**Note 1:** The rating from best to worst: Excellent, Good, Fair, Poor.

**Note 2:** These applications assume that the Ethernet interface is supplied on an adapter card.

**Note 3:** This application places the Ethernet interface on the PC AT motherboard (system planar) interfaced to the system CPU bus. If the Ethernet interface is placed on the motherboard connected to an I/O bus the architectural choice is represented by the bus (like ISA).

### CONCLUSION

The correct choice of an Ethernet controller must be a careful balance of all of the design goals. In the majority of cases, the throughput of 16-bit Ethernet controllers is more than sufficient (with the exception of servers). For 386 or greater based PC's, throughput is limited by the network operating system and the 10 Mbit/sec. data rate of Ethernet. The decision to use a 32-bit controller should be based on the need for available CPU bandwidth, and to a much lesser extent, throughput.

# Writing Drivers for the DP8390 NIC Family of Ethernet Controllers

National Semiconductor  
Application Note 874



AN-874

## INTRODUCTION

This document provides detailed information for writing drivers for the NIC Family of Ethernet Controllers; DP8390 NIC, DP83901 SNIC™, DP83902 ST-NIC™, and DP83905 AT/LANTIC™. It describes the basic components of the drivers: (1) hardware initialization, (2) initiating transmissions, and (3) servicing receive and transmit interrupts. It includes specific examples of actual network drivers (DriverInitialize, DriverSend, and DriverISR). **We recommend that you become familiar with the individual part Data-sheets.**

## HARDWARE INITIALIZATION

The initialization procedure supplies configuration parameters for the NIC Controllers to operate in the current system. This involves the CPU loading the proper values into the configuration and address registers and enabling the NIC Controllers onto the network. The following shows a list of parameters that must be initialized before the NIC Controllers become operational.

- data bus width (8 or 16 bits)
- physical address
- types of interrupts that may be serviced
- size of the Receive Buffer Ring
- FIFO threshold
- types of packets that may be received

An example of an initialization routine for a typical 8-bit system is exemplified in DriverInitialize. Note that the DATA CONFIGURATION register must be initialized before all other registers are initialized (except the COMMAND register). Note also the sequencing to enable the DP83902 and DP83905 onto the network.

## PACKET TRANSMISSION

The transmit driver is generally partitioned into two parts. The first part (DriverSend) initiates a transmission whenever the upper level software passes a packet to the driver. If the driver is unable to transmit the packet immediately (i.e., the transmitter is busy), the supplied packet is queued in a transmit-pending buffer. After initiating or queuing up the packet, DriverSend returns.

DriverSend operates in conjunction with an interrupt service routine (DriverISR). After completing the transmission, the NIC Controllers interrupt the CPU to signal the end of the transmission and indicate status information in the TRANSMIT STATUS register.

## RECEIVE DRIVER

The responsibility of the receive driver is to transfer data from the Receive Buffer Ring to the host's memory. Ideally, this process is done as fast as possible to eliminate any bottlenecks that may be incurred by the driver. The NIC Controllers facilitate removing data from the Ring by providing a Remote DMA channel to transfer data from the Ring to an I/O port which is readable by the host system. It also

maintains two pointers to track packets in the Ring: BOUNDARY and CURRENT. These registers respectively point to the last unread packet in the Ring and the next vacant location in memory to receive another packet. Generally, the receive driver removes the next packet pointed to by BOUNDARY, then increments BOUNDARY to the succeeding packet indicated by the Next Page Pointer in the 4-byte NIC Controllers receive header. This process continues until all packets have been removed from the Ring.

The NIC Controllers automate packet removal with the "send packet" command. When this command is issued, the NIC Controllers automatically load the DMA start address with BOUNDARY, load the DMA byte count from the 4-byte receive header, then begin transferring data. At the end of the DMA, the NIC Controllers update BOUNDARY with the Next Page Pointer from the receive header. To remove all packets from the Ring, the receive driver simply issues the "send packet" command until the BOUNDARY and CURRENT registers are equal.

Because of the asynchronous nature of reception, the receive driver must be interrupt driven. Typically, packet reception is given high priority since delaying packet removal may overflow the Receive Buffer Ring. If several packets in the ring have been queued, all packets should be removed in one process (i.e., a software loop which empties the Ring). In heavy traffic conditions, local memory can fill up quickly so it is important that the Ring be large enough to handle these situations.

To find out how many packets are lost due to Ring overflows or network errors, the NIC Controllers have three statistical registers to monitor the network; FRAME ALIGNMENT ERROR tally, CRC ERROR tally, and FRAMES LOST tally. These registers are useful in initially determining the size of the Ring and how many packets are lost due to network related errors (CRC errors and/or frame alignment errors).

## EXAMPLE DRIVERS

The following transmit and receive drivers are written in assembly for fast execution. The transmit driver is partitioned into two parts, DriverSend and DriverISR, while the receive driver resides entirely within DriverISR. This section gives an overview of DriverISR, followed by a description on how receive and transmit interrupts interact with DriverISR.

### Interrupt Service Routine (DriverISR)

DriverISR is concerned with interrupts originating from receptions, transmissions, and errored transmissions. Errored receptions are ignored since these are usually collision fragments and are of no use to the upper layer software. DriverISR (Figure 2) consists of (1) a packet transmitted routine and (2) a packet received routine. The basic functions of the routines are as follows:

Packet Transmitted Routine: checks the status of all transmissions and transmits the next packets in the transmit-pending queue.

1

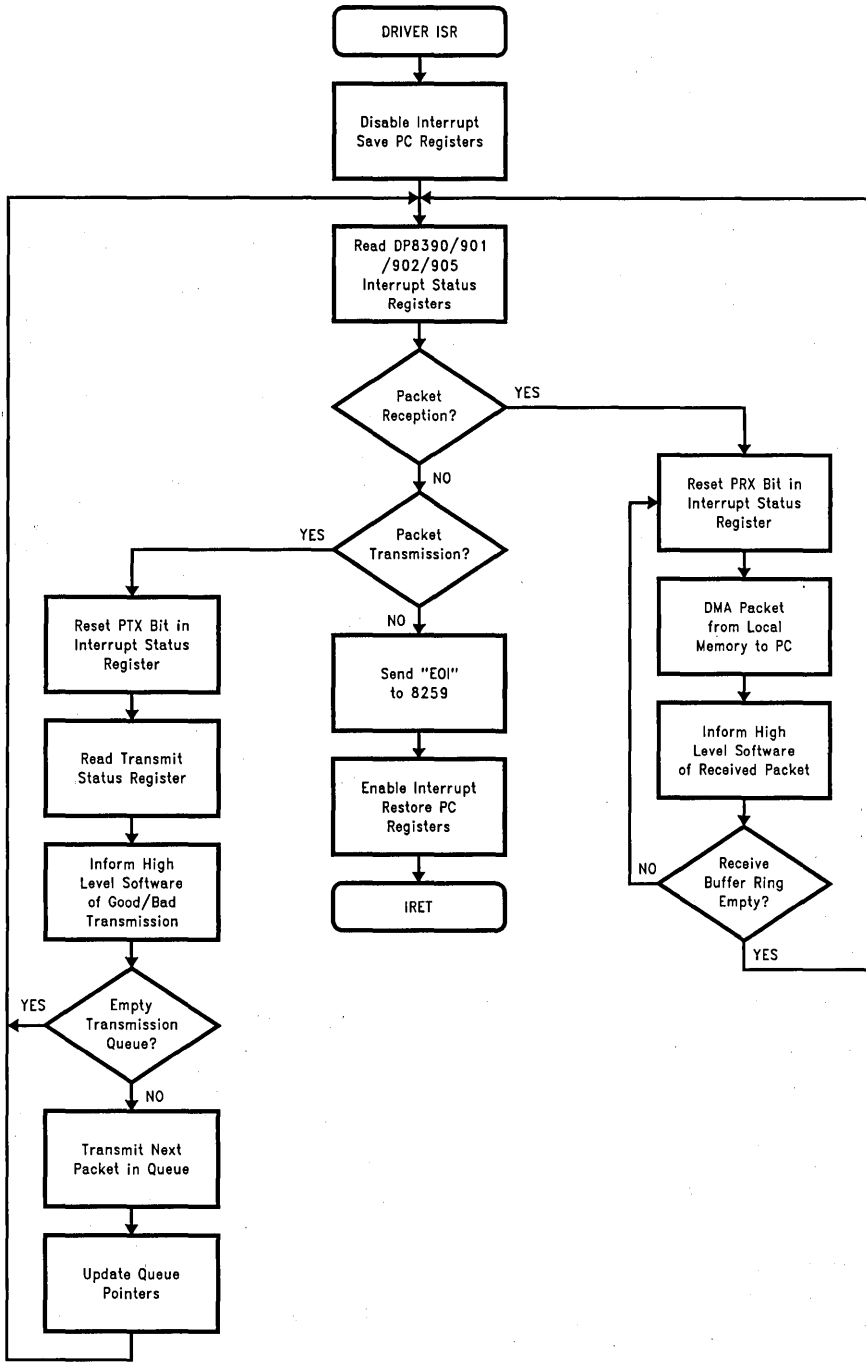
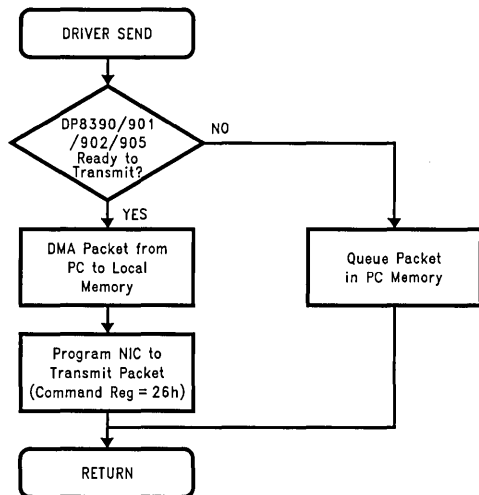


FIGURE 2. Interrupt Service Routine

Packet Received Routine: removes all packets in the receive buffer ring by using the "send packet" command of the NIC Controllers.

### Transmit Driver

The transmit drivers consist of two parts. The first part, DriverSend (Figure 3), initiates transmission when called by the upper layer software. DriverSend checks if the NIC Controllers are ready to transmit by reading the COMMAND register (TXP bit is zero). If ready, the DriverSend using the DP8390's Remote DMA channel, transfers from the PC's memory to local memory, then issues the transmit command and returns. Otherwise, if the NIC Controllers are busy (TXP bits equal one) DriverSend queues the packet in the transmit-pending queue, then returns.



TL/F/11785-3

FIGURE 3. Driver Send Routine

After a transmission is completed, DriverISR services the interrupt from the NIC Controllers and (1) reports status information by reading the TRANSMIT STATUS register and (2) transmits the next packet in the transmit-pending queue,

if any. Thus, for a transmit interrupt, DriverISR executes the following steps:

1. Reset PTX bit in INTERRUPT STATUS register.
2. Check for good transmission by reading the TRANSMIT STATUS Register.
3. If there are more packets in the transmit-pending queue, transmit the next packet; otherwise go to 4.
4. Read INTERRUPT STATUS register for any pending interrupts.

### Receiver Driver

Since the receiver driver must be interrupt driven, it resides completely within the DriverISR. When the receive interrupt occurs, one or more packets may be buffered into the Ring by the NIC Controllers. The DriverISR removes packets from the Ring and then passes them up to the host. Using the "send packet" command, packets are removed until the Ring is empty, that is, when CURRENT and BOUNDARY registers are equal. The sequence of the receive packet routine is shown below.

1. Reset the PRX bit in the INTERRUPT STATUS register.
2. Remove the next packet in the receive buffer using the "send packet" command.
3. Check to see if the receive buffer ring is empty: BOUNDARY register = CURRENT PAGE register
4. If the Ring is not empty, go to 1; otherwise read INTERRUPT STATUS register for any more pending interrupts.

### OTHER SOFTWARE CONSIDERATIONS

The NIC Ethernet Controllers require some special software considerations to operate in all network environments. In particular, the handling of overflow of the receive buffer ring must be handled EXACTLY as described in the data sheet and Design Tips.

The most efficient manner to remove packets from the transmit-pending queue is to use Driver Send to initiate transmission of the very first packet in the queue; then upon completion, use the DriverISR to transmit the remaining packets. Using this method, the DriverISR examines the queue, transmits the next available packet, then exits. The DriverISR transmits the next packet after the NIC Controllers issue the next transmit interrupt.

```

;*****
DriverInitialize

; Initializes the NIC for a typical network system.
; Receive Buffer Ring = 2800h to 4000h
; Transmit Buffer    = 2000h to 2600h
;
;      Entry: none
;*****
;*****Equates for NIC Registers*****

COMMAND          equ 300h
PAGESTART        equ COMMAND+1
PAGESTOP         equ COMMAND+2
BOUNDARY         equ COMMAND+3
TRANSMITSTATUS   equ COMMAND+4
TRANSMITPAGE     equ COMMAND+4
TRANSMITBYTECOUNT0 equ COMMAND+5
NCR              equ COMMAND+5
TRANSMITBYTECOUNT1 equ COMMAND+6
INTERRUPTSTATUS  equ COMMAND+7
CURRENT          equ COMMAND+7           ;in page 1
REMOTESTARTADDRESS0 equ COMMAND+8
CRDMAO          equ COMMAND+8
REMOTESTARTADDRESS1 equ COMMAND+9
CRDMA1          equ COMMAND+9
REMOTEBYTECOUNT0 equ COMMAND+0ah
REMOTEBYTECOUNT1 equ COMMAND+0bh
RECEIVESTATUS   equ COMMAND+0ch
RECEIVECONFIGURATION equ COMMAND+0ch
TRANSMITCONFIGURATION equ COMMAND+0dh
FAE_TALLY       equ COMMAND+0dh
DATACONFIGURATION equ COMMAND+0eh
CRC_TALLY       equ COMMAND+0eh
INTERRUPTMASK   equ COMMAND+0fh
MISS_PKT_TALLY  equ COMMAND+0fh

PSTART          equ 46h
PSTOP           equ 80h

CGroup group Code
Code segment para public 'Code'
    assume cs:CGroup, ds:CGroup, es:nothing, ss:nothing

rcr db 0 ;value for Recv config. reg
tcr db 0 ;value for trans. config. reg
dcr db 58h ;value for data config. reg
imr db 0bh ;value for intr. mask reg

```



```

;*****
DriverInitialize proc near
    public      DriverInitialize

    mov     al,21h                ;stop mode
    mov     dx,COMMAND
    out     dx,al

    mov     al,dcr
    mov     dx,DATACONFIGURATION  ;data configuration register
    out     dx,al

    mov     dx,REMOTEBYTECOUNT0
    xor     al,al
    out     dx,al                ;low remote byte count
    mov     dx,REMOTEBYTECOUNT1
    out     dx,al                ;high remote byte count

    mov     al,rcr
    mov     dx,RECEIVECONFIGURATION ;receive configuration register
    out     dx,al

    mov     al,20h
    mov     dx,TRANSMITPAGE        ;transmit page start
    out     dx,al

    mov     al,02
    mov     dx,TRANSMITCONFIGURATION ;temporarily go into Loopback mode
    out     dx,al

    mov     al,26h
    mov     dx,PAGESTART           ;page start
    out     dx,al

    mov     dx,BOUNDARY           ;boundary register
    out     dx,al
    mov     al,40h

    mov     dx,PAGESTOP           ;page stop
    out     dx,al

    mov     al,61h                ;go to page 1 registers
    mov     dx,COMMAND
    out     dx,al

    mov     al,26h
    mov     dx,CURRENT            ;current page register
    out     dx,al

    mov     al,22h                ;back to page 0, start mode
    mov     dx,COMMAND
    out     dx,al

    mov     al,0ffh
    mov     dx,INTERRUPTSTATUS    ;interrupt status register
    out     dx,al

    mov     al,imr
    mov     dx,INTERRUPTMASK      ;interrupt mask register
    out     dx,al

    mov     dx,TRANSMITCONFIGURATION
    mov     al,ter
    out     dx,al                ;TCR in normal mode, NIC is now
                                ;ready for reception

    ret
DriverInitialize endp

Code ends
end

```

```

;*****
;
; DriverSend
;
; Either transmits a packet passed to it or queues up the
; packet if the transmitter is busy (COMMAND register = 26h).
; Routine is called from upper layer software.
;
; Entry: ds:si => packet to be transmitted
; cx => byte count of packet
;*****
;*****Equates for NIC Registers*****
COMMAND equ 300h
PAGESTART equ COMMAND+1
PAGESTOP equ COMMAND+2
BOUNDARY equ COMMAND+3
TRANSMITSTATUS equ COMMAND+4
TRANSMITPAGE equ COMMAND+4
TRANSMITBYTECOUNT0 equ COMMAND+5
NCR equ COMMAND+5
TRANSMITBYTECOUNT1 equ COMMAND+6
INTERRUPTSTATUS equ COMMAND+7
CURRENT equ COMMAND+7 ;in page 1
REMOTESTARTADDRESS0 equ COMMAND+8
CRDMA0 equ COMMAND+8
REMOTESTARTADDRESS1 equ COMMAND+9
CRDMA1 equ COMMAND+9
REMOTEBYTECOUNT0 equ COMMAND+0ah
REMOTEBYTECOUNT1 equ COMMAND+0bh
RECEIVESTATUS equ COMMAND+0ch
RECEIVECONFIGURATION equ COMMAND+0ch
TRANSMITCONFIGURATION equ COMMAND+0dh
FAE_TALLY equ COMMAND+0dh
DATACONFIGURATION equ COMMAND+0eh
CRC_TALLY equ COMMAND+0eh
INTERRUPTMASK equ COMMAND+0fh
MISS_PKT_TALLY equ COMMAND+0fh
IOPORT equ COMMAND+10h

FSTART equ 46h
FSTOP equ 80h
TRANSMITBUFFER equ 40h

```

```
.CODE
```

```

DriverSend proc near
public DriverSend
cli ;disable interrupts
mov dx,COMMAND
in al,dx ;read NIC command register
cmp 26h ;transmitting?
je QueueIt ;if so, queue packet

```

```
push    cx                ;store byte count
mov     ah,TRANSMITBUFFER
xor     al,al             ;set page to transfer packet to
call   PCtoNIC           ;transfer packet to NIC buffer RAM
mov     dx,TRANSMITPAGE
mov     al,TRANSMITBUFFER
out     dx,al            ;set NIC transmit page
pop     cx                ;get byte count back
mov     dx,TRANSMITBYTECOUNT0
mov     al,cl
out     dx,al            ;set transmit byte count 0 on NIC
mov     dx,TRANSMITBYTECOUNT1
mov     al,ch
out     dx,al            ;set transmit byte count 1 on NIC
mov     dx,COMMAND
mov     al,26h
out     dx,al            ;issue transmit to COMMAND register
jmp     Finished
```

QueueIt:

```
call Queue_packet
```

Finished:

```
sti                ;enable interrupts
```

```
ret
```

```
ret
```

DriverSend

```
endp
```

```

*****
;
;                               PCtoNIC
;
; This routine will transfer a packet from the PC's RAM
; to the local RAM on the NIC card.
;
;   assumes: ds: si = packet to be transferred
;            cx     = byte count
;            ax     = NIC buffer page to transfer to
*****
public  __PCtoNIC
PCtoNIC proc  far
    push    ax                ; save buffer address
    inc     cx                ; make even
    and     cx,0fffh
    mov     dx,REMOTEBYTECOUNT0 ; set byte count low byte
    mov     al,cl
    out     dx,al
    mov     dx,REMOTEBYTECOUNT1 ; set byte count high byte
    mov     al,ch
    out     dx,al
    pop     ax                ; get our page back
    mov     dx,REMOTESTARTADDRESS0
    out     dx,al            ; set as lo address
    mov     dx,REMOTESTARTADDRESS1
    mov     al,ah            ; set as hi address
    out     dx,al
    mov     dx,COMMAND
    mov     al,12h           ; write and start
    out     dx,al
    mov     dx,IOPORT
    shr     cx,1             ; need to loop half as many times
Writing_Word:
    lodsw
    out     dx,ax            ;because of word-wide transfers
                                ;load word from ds:si
                                ;write to IOPORT on NIC board
    loop   Writing_Word
    mov     cx,0
    mov     dx,INTERRUPTSTATUS
CheckDMA:
    in     al,dx
    test   al,40h           ; dma done ???
    jnz   toNICEND         ; if so, go to NICEND
    jmp   CheckDMA         ;loop until done
toNICEND:
    mov     dx,INTERRUPTSTATUS
    mov     al,40h         ;clear DMA interrupt bit in ISR
    out     dx,al
    cld
    ret
PCtoNIC    endp

```

```

;*****
;
;                               NICtoPC
;
; This routine will transfer a packet from the RAM
; on the NIC card to the RAM in the PC.
;
;   assumes: es: di = packet to be transferred
;           cx      = byte count
;           ax      = NIC buffer page to transfer from
;*****
public  _NICtoPC
_NICtoPC proc  far
    push  ax                ; save buffer address
    inc   cx                ; make even
    and   cx,0ffffh
    mov   dx,REMOTEBYTECOUNT0
    mov   al,cl
    out   dx,al
    mov   dx,REMOTEBYTECOUNT1
    mov   al,ch
    out   dx,al
    pop   ax                ; get our page back
    mov   dx,REMOTESTARTADDRESS0
    out   dx,al            ; set as low address
    mov   dx,REMOTESTARTADDRESS1
    mov   al,ah
    out   dx,al            ; set as hi address
    mov   dx,COMMAND
    mov   al,0ah           ; read and start
    out   dx,al
    mov   dx,IOPORT
    shr   cx,1             ; need to loop half as many times
                                ;because of word-wide transfers
Writing_Word:
    in    ax,dx
    stow  Reading_Word    ;read word and store in es:di
    mov   dx,INTERRUPTSTATUS
CheckDMA:
    in    al,dx
    test  al,40h
    jnz  ReadEnd
    jmp  CheckDMA
ReadEnd:
    out   dx,al            ; clear RDMA bit in NIC ISR
    ret
_NICtoPC endp

```

```

;*****
;
; DriverISR
;
; This interrupt service routine responds to transmit, transmit error, and
; receive interrupts (the PTX, TXE, and PRX bits in the INTERRUPT STATUS
; register) produced from the NIC. Upon transmit interrupts, the upper
; layer software is informed of successful or erroneous transmissions;
; upon receive interrupts, packets are removed from the Receive Buffer
; Ring (in local memory) and transferred to the PC.
;*****
;*****Equates for NIC Registers*****
COMMAND equ 300h
PAGESTART equ COMMAND+1
PAGESTOP equ COMMAND+2
BOUNDARY equ COMMAND+3
TRANSMITSTATUS equ COMMAND+4
TRANSMITPAGE equ COMMAND+4
TRANSMITBYTECOUNT0 equ COMMAND+5
NCR equ COMMAND+5
TRANSMITBYTECOUNT1 equ COMMAND+6
INTERRUPTSTATUS equ COMMAND+7
CURRENT equ COMMAND+7 ;in page 1
REMOTESTARTADDRESS0 equ COMMAND+8
CRDMA0 equ COMMAND+8
REMOTESTARTADDRESS1 equ COMMAND+9
CRDMA1 equ COMMAND+9
REMOTEBYTECOUNT0 equ COMMAND+0ah
REMOTEBYTECOUNT1 equ COMMAND+0bh
RECEIVESTATUS equ COMMAND+0ch
RECEIVECONFIGURATION equ COMMAND+0ch
TRANSMITCONFIGURATION equ COMMAND+0dh
FAE_TALLY equ COMMAND+0dh
DATACONFIGURATION equ COMMAND+0eh
CRC_TALLY equ COMMAND+0eh
INTERRUPTMASK equ COMMAND+0fh
MISS_PKT_TALLY equ COMMAND+0fh

PSTART equ 46h
PSTOP equ 80h

CGroup group Code
Code segment para public 'Code'
assume cs:CGroup, ds:CGroup, es:nothing, ss:nothing
; External routines
extrn DriverSend: near
byte_count dw ?
imr db lbh ;image of Interrupt Mask register

```

```

;*****
; Begin of Interrupt Service Routine
;*****
netisr proc near
    public netisr
    cli
    push ax           ;save regs
    push bx
    push cx
    push dx
    push di
    push si
    push ds
    push es
    push bp
    mov al,0bch
    out 2lh,al       ;turn off IRQ3
    sti
    mov ax,CGroup
    mov ds,ax        ;ds=cs

;*****
; Read INTERRUPT STATUS REGISTER for receive packets, transmitted
; packets and errored transmitted packets.
;*****

poll:
    mov dx,INTERRUPTSTATUS
    in  al,dx
    test al,1        ;packet received?
    jnz pkt_recv_rt
    test al,0ah      ;packet transmitted?
    jz  exit_isr     ;no, let's exit
    jmp pkt_tx_rt

exit_isr:
    mov dx,INTERRUPTMASK ;disabling NIC's intr
    mov al,0
    out dx,al
    cli
    mov al,0b4h       ;turn IRQ3 back on
    out 2lh,al
    mov al,63h        ;send 'EOI' for IRQ3
    out 20h,al
    sti

    mov dx,INTERRUPTMASK ;NOTE: intr from the NIC
    mov al,imr         ; are enabled at this point so
    out dx,al          ; that the 8259 interrupt
                        ; controller does not miss any
                        ; IRQ edges from the NIC
                        ; (IRQ is edge sensitive)

    pop bp
    pop es
    pop ds
    pop si
    pop di
    pop dx
    pop cx
    pop bx
    pop ax
    iret

```

```

;*****
; Packet Receive Routine (pkt_rcv_rt) - clears out all good
; packets in local receive buffer ring. Bad packets are ignored.
;*****

pkt_rcv_rt:
    mov dx,INTERRUPTSTATUS
    in  al,dx
    test al,10h           ;test for a Ring overflow
    jnz ring_ovfl
    mov al,1
    out dx,al           ;reset PRX bit in ISR
    mov ax,next_packet
    mov cx,packet_length
    mov es,seq_rcv_pc_buff
    mov di,offset_rcv_pc_buff
    NICToPC

;*****
;
; Inform upper layer software of a received packet to be processed
;
;*****
; checking to see if receive buffer ring is empty
check_ring:
    mov dx,BOUNDARY
    in  al,dx
    mov ah,al           ;save BOUNDARY in ah
    mov dx,COMMAND
    mov al,62h
    out dx,al           ;switched to pg 1 of NIC
    mov dx,CURRENT
    in  al,dx
    mov bh,al           ;bh = CURRENT PAGE register
    mov dx,COMMAND
    mov al,22h
    out dx,al           ;switched back to pg 0
    cmp ah,bh           ;rcv buff ring empty?
    jne pkt_rcv_rt
    jmp poll

;*****
;
; The following code is required to recover from a Ring overflow.
; See Sec. 2.0 of datasheet addendum.
;
;*****

ring_ovfl:
    mov dx,COMMAND
    mov al,21h
    out dx,al           ;put NIC in stop mode

    mov dx,REMOTEBYTECOUNT0
    xor al,al
    out dx,al
    mov dx,REMOTEBYTECOUNT1
    out dx,al

    mov dx,_INTERRUPTSTATUS
    mov cx,7fffh       ;load time out counter

```



```

wait_for_stop:
    in    al,dx
    test  al,80h                ;look for RST bit to be set
    loop wait_for_stop         ; if we fall thru this loop, the RST bit may not get
                                ; set because the NIC was currently transmitting

    mov  dx,TRANSMITCONFIGURATION
    mov  al,2
    out  dx,al                 ;into loopback mode 1
    mov  dx,COMMAND
    mov  al,22h
    out  dx,al                 ;into stop mode

    mov  ax,next_packet
    mov  cx,packet_length
    mov  es,seg recv_pc_buff
    mov  di,offset recv_pc_buff
    NICtoPC

    mov  dx,INTERRUPTSTATUS
    mov  al,10h
    out  dx,al                 ;clear Overflow bit

    mov  dx,TRANSMITCONFIGURATION
    mov  al,tcr
    out  dx,al                 ;put TCR back to normal mode
    jmp  check_ring

;*****
;
; packet_transmit_routine (pkt_tx_rt) -determine status of
; transmitted packet, then checks the transmit-pending
; queue for the next available packet to transmit.
;
;*****

pkt_tx_rt:
    mov  dx,INTERRUPTSTATUS
    mov  al,0ah
    out  dx,al                 ;reset PTX and TXE bits in ISR

    mov  dx,TRANSMITSTATUS     ;check for erroneous TX
    in   al,dx
    test al,38h                ;is FU, CRS, or ABT bits set in TSR
    jnz  bad_tx

;*****
; Inform upper layer software of successful transmission
;*****

    jmp  chk_tx_queue         ;in here if bad TX

bad_tx:
;*****
;
; Inform upper layer software of erroneous transmission
;
;*****

```

```
chk_tx_queue:
    call  Check_Queue      ; see if a packet is in queue
                                ; assume Check_Queue will a non-zero
    cmp  cx,0             ; value in cx and pointer to the
    je   poll             ; packet in DS:SI if packet is
    call DriverSend       ; available. Returns cx = 0 otherwise
    jmp  poll
netisr endp
```

# DP83905EB-AT AT/LANTIC™ Evaluation Board

National Semiconductor  
Application Note 875  
Rick Willardson



## 1.0 OVERVIEW

The DP83905EB-AT AT/LANTIC Demonstration board provides system designers with complete 16-bit 10BASE-T, 10BASE2 and 10BASE5 Ethernet Solutions in a half-size, jumperless, ISA adapter card. The board uses only four ICs and can be configured as either a shared memory or an I/O port adapter card. All of the bus interface logic is implemented in the DP83905, AT Local Area Network Twisted Pair Interface Controller (AT/LANTIC). The AT/LANTIC uses an E<sup>2</sup>PROM to store the board's IEEE node address and its own configuration information. It uses two 8k x 8 SRAMs for buffering packets to and from the network.

The DP83905 has a built in Manchester Encoder/Decoder and Twisted Pair Transceiver. This allows the AT/LANTIC to transmit and receive packets on a 10BASE-T network with the addition of only a filter and some discrete components. An AUI interface on the AT/LANTIC can also be used to run a 10BASE5 or 10BASE2 network with the addition of two 1:1 pulse transformers, a DC-DC Converter and the DP8392 Coaxial Transceiver Interface (CTI). Refer to the schematic at the end of this Application Note.

## 2.0 ARCHITECTURAL FEATURES

- Designed with the DP8390 Network Interface Controller, NIC
- Complete Ethernet solution with only 4 ICs
- Board options are configurable in software
- Half-size PC-AT® adapter card
- 2 Modes for ISA interface—Shared Memory or I/O Port (NE2000<sup>plus</sup>™ compatible)
- 10BASE-T, 10BASE2 or 10BASE5 connectivity
- Serial EEPROM stores IEEE address and AT/LANTIC configuration while using less power than a typical bipolar PROM
- Surface mount technology on most parts
- Boot PROM socket to allow diskless boot from NetWare™, LANManager and other network operating systems
- Able to select one of eight interrupts

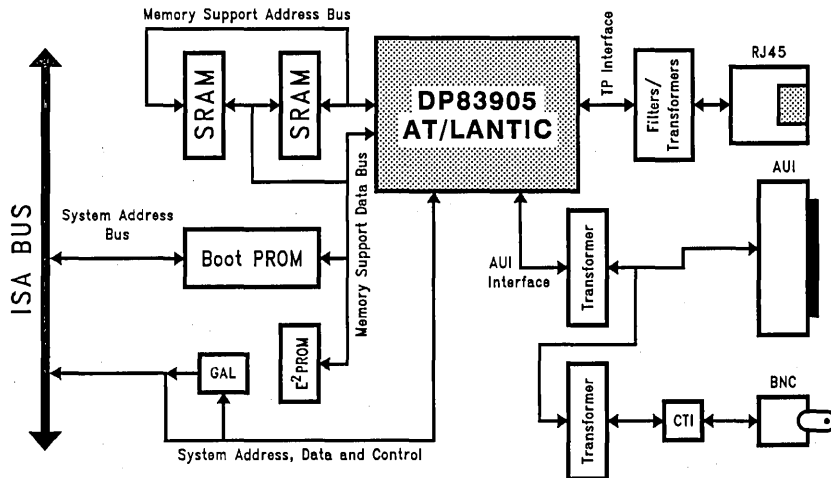


FIGURE 1. DP83905EB-AT Block Diagram

TL/F/11786-1

### 3.0 BOARD DESIGN AND LAYOUT

Figure 2 shows the part placement and power plane layout for the DP83905EB-AT Demonstration Board. The design is straight forward because all of the bus interface logic, including the 24 mA ISA bus drivers, is internal to the AT/LANTIC. The Twisted Pair interface is simple as the TPI transceiver is also internal to the AT/LANTIC. Thin Ethernet (10BASE2) is achieved with the addition of a DP8392CV CTI device and a DC-DC converter.

#### 3.1 ISA Bus Interface

All of the ISA Bus interconnects of the AT/LANTIC can be directly routed to the ISA Bus connector. Changing the physical placement of the AT/LANTIC may make these traces easier to route, but in order to keep the Twisted Pair Interface traces as short as possible, the AT/LANTIC should be placed as close as possible to the card's metal bracket. This makes the routing of the ISA Bus traces longer and hence noisier. To filter low frequency noise (kHz range) on the bus a 22  $\mu$ F decoupling capacitor was placed near the bus connector between power and ground.

The Boot PROM address lines also come directly from the ISA Bus connector. The reason for this is that the AT/LANTIC's Memory Support Address Bus does not buffer address bit A0 which is needed by the boot PROM (the AT/LANTIC only does word aligned transfers when in 16-bit mode). Boot PROM addresses must come directly off the ISA Bus where byte or word aligned transfers may be used by software. The physical location of the PROM is not critical.

#### 3.2 Memory Support Bus Interface

The Memory Support Interface is that part of the design which is used by the AT/LANTIC's Local DMA and Auto-Configuration. This includes the SRAMs and the E<sup>2</sup>PROM.

Figure 1 shows how both SRAMs and the E<sup>2</sup>PROM are connected to the Memory Support Bus of the AT/LANTIC. The AT/LANTIC uses the SRAM (8k x 16) for buffering transmit and receive packets and it uses the E<sup>2</sup>PROM to store the IEEE Node Address, a checksum, a board type, Configuration Registers A, B and C and codes which determine the data width in which the board is to run. Primarily the E<sup>2</sup>PROM is used on start up; once the AT/LANTIC reads the configuration data and the IEEE node address out of the E<sup>2</sup>PROM, this information is stored in the AT/LANTIC and made accessible by software.

In order to change the boot configuration of the board, the current configuration is changed in software and the AT/LANTIC downloads the new configuration to the E<sup>2</sup>PROM. The new configuration will be stored so that when the board is powered up again, the new configuration will be loaded. (See Section 4.0—older designs required hardware jumpers to do this). The E<sup>2</sup>PROM that the AT/LANTIC uses (NM93C06) is a serial device. The AT/LANTIC uses its MSD0–2 pins for Serial Data Out, Serial Data In and Clock to the E<sup>2</sup>PROM when reading or writing it.

The two RAMs provide 16 kbytes of memory for the AT/LANTIC to use for buffering received packets and for the system to use for buffering transmit packets. In Shared RAM mode the RAM logically resides in system memory, but it must still be connected to the AT/LANTIC's Memory Support Bus. The AT/LANTIC buffers the data and address from the ISA Bus, decodes the address and drives chip select and the read or write strobe to the RAM. Note that the board was designed to accommodate either DIP or SOP SRAMs.

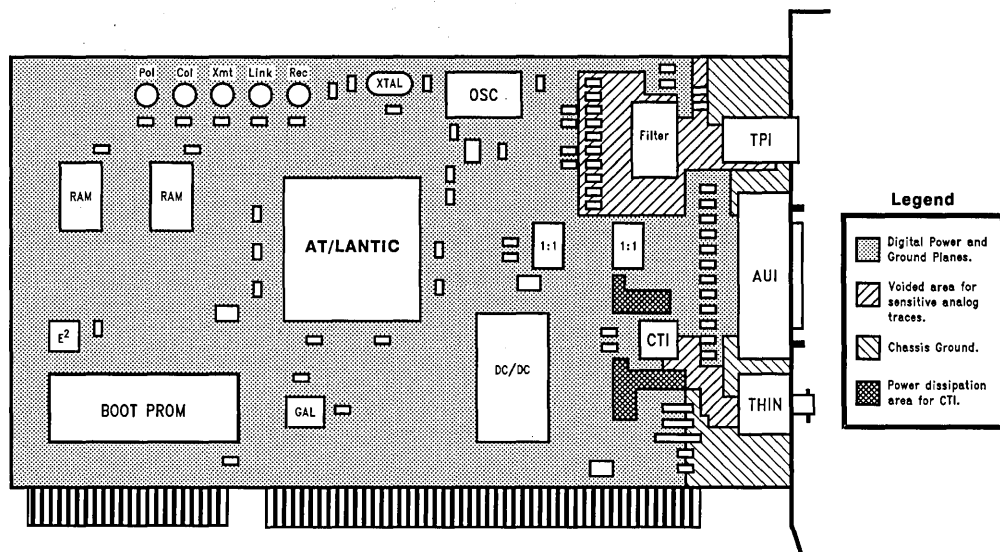


FIGURE 2. DP83905EB-AT Board Layout and Component Placement

TL/F/11786-2

### 3.3 AT/LANTIC

Some design and layout considerations were made with respect to the DP83905. These are described in the following three sections.

#### 3.3.1 Crystal and Oscillator Design

The AT/LANTIC has been designed to operate with either a crystal or an oscillator module. The DP83905EB-AT comes assembled with the crystal option. If an oscillator module is used, the crystal and the load capacitors C1 and C2 must be removed from the board. If the capacitors are not removed, they will create excessive loading on the clock which may stop the card from working. When mounting an oscillator, it should be raised slightly off the board so that it is not touching the metal pads for the capacitors C1 and C2 (some oscillator modules have built-in insulating raisers).

There are two layout considerations with respect to the clock. First, the traces for the clock should be short and the crystal or oscillator should be placed close to the AT/LANTIC (see *Figure 2*). Second, if a standard size crystal is being used and it must lie flat on the PC board, the power planes should be voided in that area. Note that on the DP83905EB-AT a low-profile crystal is being used that is not laying down. In this situation, it is not critical that the power planes be removed.

#### 3.3.2 Decoupling for the AT/LANTIC

The AT/LANTIC, like any other VLSI device, is composed of multiple functional/logic blocks. In some cases, these blocks run off of separate power rails internal to the part. Some of these blocks (such as the Twisted Pair Transceiver) can be quite susceptible to noise. Not only can the input signals couple noise from the board and environment, but output signals can transmit noise to the environment. Also, since the separate power pins are all connected to the same 5V supply in the PC, other noisy power signals can affect power supplies that need to be kept noise free. By separately decoupling the supply pins to the AT/LANTIC, the internal supplies are reasonably isolated from each other. Instead of placing decoupling capacitors near the AT/LANTIC, place the decoupling capacitors directly to the power pins. Each of the seven blocks within the AT/LANTIC should be decoupled by at least one 0.01  $\mu\text{F}$  capacitor. Thus, each of the power rails in the chip has independent decoupling. This minimizes EMI and reduces power supply noise.

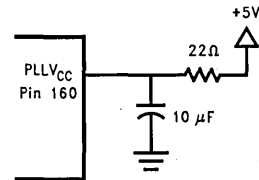
Another area of concern is the AT/LANTIC's ISA bus interface. This section is driving the ISA bus with 24 mA bus drivers. Not only is the current high, but the rise and fall times of the bus signals are fast. By decoupling the ISA interface supply, noise produced by these drivers is reduced.

Decoupling of noise from the PC bus is achieved using 0.01  $\mu\text{F}$  and 22  $\mu\text{F}$  capacitors (used to filter out lower frequency noise in the order of a few kHz). One 22  $\mu\text{F}$  is placed near the media end of the board and the other near the ISA bus connector.

#### 3.3.3 PLL Power Supply Noise

The VCO (Voltage Controlled Oscillator) block of the receive PLL (Phase-Lock Loop) within the AT/LANTIC is sensitive to noise in the frequency range of 10 kHz to 400 kHz. As little as 100 mV of noise in this range can cause the PLL

to lose lock on an incoming packet. In order to improve the performance of the PLL, a single pole filter should be used on the PLL power supply pin. This is shown below in *Figure 3*.



TL/F/11786-3

FIGURE 3. PLL  $V_{CC}$  Noise Filter

#### 3.4 Twisted Pair Interface (TPI)

This section refers to the layout diagram in *Figure 2*, of this Application Note.

The Twisted Pair Interface of the DP83905EB-AT is simple: the only components needed external to the AT/LANTIC are the pre-emphasis resistors, some capacitors, a transformer/filter module and, of course, the RJ45 modular connector. Resistors R19-22 (refer to the schematic at the end of this application note) provide the preemphasis for the TXO $\pm$  output as described in the AT/LANTIC datasheet. R23 damps the TXO $\pm$  output undershoot. Resistors R24 and R25 terminate the receive Twisted Pair line. C9, C43-C45 filter out the common mode noise in the transformer/filter. C46 and C47 filter out the high frequency noise harmonics on the TXO $\pm$  output. All these components together with the transformer/filter module should be placed as close to the AT/LANTIC's twisted pair interface pins as possible. The traces should be kept straight and should not run near any other traces. The traces on the cable side of the transformer/filter module to the RJ45 connector should be kept short and straight too. Additionally, make sure that all differential traces are tracked in parallel so they are equal-length and will pick up the same amount of common mode noise.

The entire area under all of these signals should be voided of signal traces and power planes, as these could couple noise into the Tx/Rx signals. This is shown in *Figure 2*. This part of the circuit is very critical and will affect FCC test results directly. Precaution should be taken during layout to ensure that noise is reduced as much as possible.

#### 3.5 Attachment Unit Interface (AUI)

The DP83905EB-AT can be used with an external Medium Attachment Unit (MAU) so that connectivity to 10BASE5 or another medium can be achieved. This interface is via the 15-pin D-type connector and is a direct interface to the AT/LANTIC's ENDEC. The AT/LANTIC's ENDEC requires two 270 $\Omega$  pull-down resistors on the transmit outputs and 78 $\Omega$  terminations on the receive and collision inputs. Also, the RX, TX and CD lines must be AC coupled using a standard 1:1 pulse transformer. This is the only design requirement for the AUI interface: it is needed to isolate the AT/LANTIC from DC fault conditions. Although these differential signals are not as noise sensitive as the Twisted Pair inputs, care should still be taken when laying out this section of the board. It is not necessary to void the power planes under them.

### 3.6 Coaxial Transceiver Interface (CTI)

This section refers to the layout diagram in *Figure 2* of this application note.

The Coax Transceiver Interface (DP8392CV) should be placed very close to the BNC connector. This will allow the TXO/RXI trace to be short and straight. The area under this trace should be voided of all other signals and power planes (as shown in *Figure 2*) in order to meet the IEEE 802.3 input capacitance requirement. The BNC connector on the demonstration board is a standard part, however, "quiet" connectors are readily available to reduce noise for performance and FCC qualification. The CTI requires an area of copper on the top of the board for heat dissipation. This is also shown in *Figure 2* and is documented in the DP8392CV datasheet. The ground shield of the BNC connector (and of the coax cable) is resistively and capacitively decoupled to chassis ground. There is a chassis ground strip on each layer of the board running from the top to the bottom of the PC board. This is connected to the chassis of the PC through the 15-pin AUI connector's body metal. Chassis ground is then capacitively decoupled to digital ground. The chassis ground trace along the front of the board forms a "shield" so that noise is not emitted into or received from the environment.

In order to provide a jumperless solution, a 1:1 transformer is required between the CTI and the 15-pin AUI connector. This provides 500V isolation between the CTI and the DTE ground as required by the IEEE 802.3 specification.

#### 3.6.1 DC-DC Converter Solution

The DP83905EB-AT has been designed to use a standard DC-DC converter which operates from a 12V supply and meets the specifications of the DP8392CV. When the AT/LANTIC is programmed for Thin Ethernet, the THIN output is driven high to turn on the DC-DC converter. For Thick Ethernet and Twisted Pair (10BASE5/T) the AT/LANTIC drives the THIN output low, turning off the DC-DC converter.

Table I shows the possible settings for bits PHYS1 and PHYS0, the Physical Layer Interface selection bits in Configuration Register B. When programmed as [0 1] the THIN output of the AT/LANTIC will go high which will turn on the DC-DC converter. When programmed to any other combination the THIN output will go low, turning the DC-DC converter off.

TABLE I. Physical Medium Selection

PHYS1-0	Media
0 0	10BASE-T
0 1	10BASE2
1 0	AUI (10BASE5)
1 1	TPI (RSL)

#### 3.7 Interrupt Scheme

The DP83905EB-AT will support the selection of one of eight host interrupts. (Encoded mode.)

To program the AT/LANTIC to use the encoded mode interrupt scheme, set the INTMODE bit in Configuration Register C HIGH (logic 1)—see Section 4.3 for details. INT3 is the active interrupt output and INTO, 1, 2 are programmable outputs containing the values in bits 3-5 of configuration register A.

The DP83905EB-AT connects the AT/LANTIC's INTO-3 pins to the inputs of a 16V8 GAL. The GAL maps the INTO-2 inputs to the host's interrupts, as shown in *Figure 4* and Table II. This interrupt scheme is Novell NE2000plus compatible. The INT3 pin from the AT/LANTIC strobes true when an interrupt is generated by the AT/LANTIC's internal circuitry. The selected interrupt is output on the GAL (strokes LOW) for the duration of the INT3 strobe.

This means that the interrupt outputs are intended for use in an edge triggered interrupt environment. They will not function correctly in a machine with level triggered interrupts—interrupts may be missed.

By definition the ISA bus uses edge triggered interrupts, however, in an EISA machine, interrupts can be system selected to be either edge or level triggered.

```

IRQ3.OE = !INT3 & !INT2 & !INT1 & !INT0;
IRQ3 = 0;
IRQ4.OE = !INT3 & !INT2 & !INT1 & INTO;
IRQ4 = 0;
IRQ5.OE = !INT3 & !INT2 & INT1 & !INT0;
IRQ5 = 0;
IRQ9.OE = !INT3 & !INT2 & INT1 & INTO;
IRQ9 = 0;
IRQ10.OE = !INT3 & INT2 & !INT1 & !INT0;
IRQ10 = 0;
IRQ11.OE = !INT3 & INT2 & !INT1 & INTO;
IRQ11 = 0;
IRQ12.OE = !INT3 & INT2 & INT1 & !INT0;
IRQ12 = 0;
IRQ15.OE = !INT3 & INT2 & INT1 & INTO;
IRQ15 = 0;

```

FIGURE 4. GAL Contents

TABLE II. Interrupt Selection

INT 2, 1, 0	Interrupt Level
0 0 0	IRQ 3
0 0 1	IRQ 4
0 1 0	IRQ 5
0 1 1	IRQ 9
1 0 0	IRQ 10
1 0 1	IRQ 11
1 1 0	IRQ 12
1 1 1	IRQ 15

#### 3.8 Status LEDs

Five LEDs are located at the top of the board to indicate network status. The LEDs from left to right are: POLARITY (POL), COLLISION (COL), TRANSMIT (XMT), LINK (LNK) and RECEIVE (REC).

The POL LED is lit when the TPI module detects seven consecutive link pulses or three consecutive receive packets with reversed polarity.

The COL LED is lit for approximately 50 ms whenever a collision is detected.

The XMT LED is lit for approximately 50 ms whenever the AT/LANTIC controller transmits data.

In TPI mode, the LNK LED is lit while link pulses are being received. This indicates that the twisted-pair connection is active.

The REC LED is lit for approximately 50 ms whenever received data is detected.

### 3.9 Bed-of-Nails Testing

The DP83905EB-AT supports in-circuit testing to ensure that the board has been assembled correctly with working parts. The schematic for the board shows each of these points (TPxx) as test points. Physically, they are holes on the board or surface mount pads. Every node on the board that does not include a through hole part has an associated test point. The AT bus signals are also brought out to test points. This allows a bed-of-nails tester to probe every node of the board.

### 4.0 CONFIGURATION OPTIONS

Please refer to the AT/LANTIC hardware and software design guides for a more detailed explanation of this section.

The DP83905EB-AT does not require jumpers because all the configuration options for the board can be programmed by software. The serial E<sup>2</sup>PROM can also be written to by the AT/LANTIC, allowing the configuration to be changed and saved. The memory map of the E<sup>2</sup>PROM is shown in Table III.

TABLE III. E<sup>2</sup>PROM Contents

	D15-D8	D7-D0
0FH	73H	Config. Reg. C
0EH	Config. Reg. B	Config. Reg. A
0DH	Not used	Not used
0CH	Not used	Not used
0BH	Not used	Not used
0AH	Not used	Not used
09H	Not used	Not used
08H	42H	42H
07H	57H	57H
06H	Not used	Not used
05H	Not used	Not used
04H	Not used	Not used
03H	Checksum	Board Type
02H	E'Net Address 5	E'Net Address 4
01H	E'Net Address 3	E'Net Address 2
00H	E'Net Address 1	E'Net Address 0

The fields in the E<sup>2</sup>PROM are defined as:

**CONFIG REG. A**—Stores base I/O Address and Interrupt number that the board is currently using. The AT/LANTIC has write access to this location. See bit description in Section 4.1.

**CONFIG REG. B**—Stores Physical Layer Interface and programmable bus options. The AT/LANTIC has write access to this location. See bit description in Section 4.2.

**CONFIG REG. C**—Stores Boot PROM address and other hardware specific configuration options. The AT/LANTIC cannot write to this location. See bit description in Section 4.3.

**42H**—This location (08H) must contain 42H (ASCII "B"). When DWID is low, the software will read this location in the PROM Store of the AT/LANTIC and determine that the board is in an 8-bit slot. The AT/LANTIC cannot write to this location.

**57H**—This location (07H) must contain 57H (ASCII "W"). When DWID is high, the software will read this location in the PROM Store of the AT/LANTIC and determine that the board is in a 16-bit slot. The AT/LANTIC cannot write to this location.

**CHECKSUM**—This location is only used in Shared RAM mode. The checksum is used to verify the Ethernet Address and the board type. The two's complement addition of the last eight bytes (03H-00H) must equal FFH, otherwise there is an error. The value of the checksum is determined from this. The AT/LANTIC cannot write to this location.

**BOARD TYPE**—This location is only used in Shared RAM mode. The Board Type indicator tells the network driver what hardware is being used. The AT/LANTIC cannot write to this location. When the board is configured to operate in WD8013EBT compatible Shared RAM mode, the Board Type should be 05H.

**ETHERNET ADDRESS**—The last six bytes of the E<sup>2</sup>PROM contain the Ethernet Address which is issued by the IEEE. These bytes must be programmed uniquely prior to fitting the device onto the board, as the AT/LANTIC is unable to write to these locations.

#### 4.1 Configuration Register A

Please refer to the AT/LANTIC datasheet for a detailed description of the bits in Configuration Register A. The following descriptions show the bit definitions and their default settings.

**IOAD2-0 (D2-D0)**—The Base I/O Address for the DP83905EB-AT is programmed by these three bits. When programmed to [0 0 1], the AT/LANTIC will power up in software mode and not respond to any I/O Address. However, it will monitor the parallel port (278H) for four consecutive writes. On the fourth write to 278H, bus data will be loaded into Configuration Register A, setting the I/O address for the board. This is done so that the board will not conflict with other I/O slaves in the PC. The method the AT/LANTIC uses for monitoring the parallel port and activating configuration load is unique; no other device/software is likely to perform four consecutive writes to this location. Note that the DP83905EB-AT is shipped in this mode. When run the first time, the base I/O address must be changed using the ATLES software package.

**INT2-0 (D5-D3)**—Based on Configuration Register C, these three bits select which interrupt line is directly driven or which decode is used for coded interrupts. The DP83905EB-AT comes programmed to drive coded interrupt 3. [INT2-0 = 0 0 0].

**FREAD (D6)**—This enables the Fast Read option of the AT/LANTIC's Remote DMA Read operation. The DP83905EB-AT is shipped with the FREAD function disabled. [FREAD = 0].

**MEMIO (D7)**—Selects either I/O Port mode or Shared RAM mode. The DP83905EB-AT comes configured to run in I/O Port mode. [MEMIO = 0].

#### 4.2 Configuration Register B

Please refer to the AT/LANTIC Datasheet for a detailed description of the bits in Configuration Register B. The following descriptions show the bit definitions and their default settings.

**PHYS1-0 (D1-D0)**—The Physical Layer Interface bits select the type of physical layer being used on the board. This could be 10BASE-T, 10BASE2, AUI (10BASE5) or Reduced Squelch Twisted Pair. The DP83905EB-AT comes configured to use 10BASE2. [PHYS1,0 = 0 1].

**GDLNK (D2)**—This bit enables link test pulse generation and integrity checking when using twisted pair. It can also be read to indicate status. Link pulse generation and checking is disabled by writing a 1 to this bit. The DP83905EB-AT is shipped with link testing enabled.

**IO16CON (D3)**—One of two methods of generating  $\overline{IO16}$  is chosen using this bit. In normal operation,  $\overline{IO16}$  is driven off of the address decode.  $\overline{IO16}$  can be configured to be driven from the slave read or write strobe by this bit. The DP83905EB-AT, as shipped, will generate  $\overline{IO16}$  from address decode. [IO16CON = 0].

**CHRDY (D4)**—The way the AT/LANTIC drives  $\overline{IOCHRDY}$  can be selected by programming this bit. When low, the AT/LANTIC will drive  $\overline{IOCHRDY}$  after a slave strobe is asserted. When high, the AT/LANTIC will drive  $\overline{IOCHRDY}$  on BALE being asserted; this may be required when being used with some AT bus chipsets that sample  $\overline{IOCHRDY}$  early. The DP83905EB-AT, as shipped, will drive  $\overline{IOCHRDY}$  after slave strobe is asserted.

**BE (D5)**—This bit can be read by software to determine if there was a Bus Error. A bus error will occur if the AT/LANTIC attempts to insert wait states into a system access and the system terminates the cycle without inserting wait states.

**BPWR (D6)**—This bit protects Boot PROM write cycles. When high, the AT/LANTIC can generate write cycles to the Boot PROM. This feature is intended for use with writeable Boot storage devices. The DP83905-AT is not shipped with a Boot PROM, so this is programmed low.

**EELoad (D7)**—The EELoad bit enables/disables the AT/LANTIC from writing the configuration information into the E2PROM. This bit must be set before running the E2PROM load algorithm documented in the AT/LANTIC datasheet.

#### 4.3 Configuration Register C

Please refer to the AT/LANTIC Datasheet for a detailed description of all the bits in Configuration Register C. The following descriptions show the bit definitions and their default settings.

**BPS3-0 (D3-D0)**—These four bits select the memory address and size of the Boot PROM. If BPS3-0 is equal to [0 0 0 X], the Boot PROM is disabled. The DP83905EB-AT comes configured to operate without the boot PROM.

**COMP**—When this bit is programmed high, the AT/LANTIC's memory uses the full 64 kbytes of RAM. When low, the memory map is compatible with either the NE2000*plus* or the WD8013EBT (16 kbytes of RAM). The DP83905EB-AT is configured to run in compatible mode.

**INTMODE**—This bit selects which mode the AT/LANTIC's interrupt will run. When low, direct drive interrupts are used. When high, coded interrupts are used. The DP83905 is configured to use coded interrupts.

**CLKSEL**—When low, the NIC core of the AT/LANTIC is clocked by the 20 MHz clock on the X1 input. When high, an external clock other than 20 MHz can be used to clock BSCK. The DP83905EB-AT is configured to run on the internal 20 MHz clock.

**SOFEN**—This bit enables the software to update configuration registers A and B. When high, the configuration registers are not accessible by software. The DP83905EB-AT is configured to allow software to update the configuration registers.

#### 5.0 FUNCTIONAL OPERATION

The DP83905EB-AT takes advantage of the AT/LANTIC's ability to function in either Shared RAM mode or I/O Port mode. When in Shared RAM mode, the AT/LANTIC's bus interface is configured such that the local RAM on the board is mapped into system memory as well as the AT/LANTIC's memory. This allows the driver to have direct access to the AT/LANTIC's local memory. In I/O Port mode, the local RAM on the board is only mapped in the AT/LANTIC's address space. This permits the AT/LANTIC to have sole ownership of the RAM. The network driver has access to the RAM through a data latch, which is in I/O space.

In both architectures, the AT/LANTIC uses the local RAM to buffer both transmit and receive packets. During transmissions, the driver will write Ethernet packets into a designated block in the RAM, typically called the Transmit Buffer. In most cases, the transmit buffer is large enough for only one packet, and although they can be, packets are not queued for transmission. When the entire packet is written to memory and the AT/LANTIC is programmed to perform a transmission, the AT/LANTIC will begin reading the packet and storing it in blocks of bytes or words into its FIFO. From the FIFO, the data is serialized, encoded and transmitted to the network.

In a like manner, the AT/LANTIC uses the local RAM to store packets as they are received from the network. The AT/LANTIC's Receive Buffer is organized as a ring (or FIFO) so that multiple packets can be buffered and at the same time the network driver can read packets that have already been received. As packets are received into the AT/LANTIC's FIFO from the network, the AT/LANTIC's DMA puts the packet data into the local RAM. After packets have been buffered by the AT/LANTIC, the network driver will read the packets out of the buffer ring. Sections 5.1 and 5.2 describe how the AT/LANTIC operates in each of its bus modes. Sections 5.3 and 5.4 briefly describe the transmit and receive operations.

##### 5.1 Shared RAM Mode

With the AT/LANTIC configured to operate in Shared RAM Mode, the DP83905EB-AT is hardware compatible with a WD8013EBT card. Changes to the EEPROM contents are needed for the WD8013EBT drivers to run.



While operating in Shared RAM mode, the local RAM on the DP83905EB-AT can be mapped into system memory at one of several different locations. The upper and lower address decode and memory width can be selected by the Shared Memory Control Registers of the AT/LANTIC. The depth of the memory can be selected in Configuration Register C by setting the compatible (COMP) bit. This will configure the RAM to be either 16 kbytes (compatible with the NE2000*plus* and WD8013EBT) or 64 kbytes. The local buffer RAM is then accessible to both the AT/LANTIC and the ISA bus. An arbiter, internal to the AT/LANTIC, will determine which device currently has access to the RAM. The AT/LANTIC will only access the RAM during local DMA burst cycles for transmit and receive operations, however, it is critical that the AT/LANTIC is granted the bus within a short time after requesting it. For this reason, the arbiter will grant access to the internal NIC core in the event that both the AT/LANTIC and the host request the local bus at the same time. This arbitration is transparent to both the hardware and the network driver.

### 5.2 I/O Port Mode

If the AT/LANTIC is configured for I/O Port Mode, the DP83905EB-AT will be hardware compatible with an NE2000*plus* card. The NE2000*plus* network drivers will then run on the DP83905EB-AT.

In I/O Port mode, the AT/LANTIC is configured to use the Remote DMA function of the NIC core. In this mode of operation, the local RAM of the DP83905EB-AT is accessible only by the AT/LANTIC. In order for the driver to read any packets from the buffer ring or write any packets into the transmit buffer, the AT/LANTIC must be programmed to perform a remote DMA cycle. To execute a Remote DMA, the bus interface logic of the AT/LANTIC performs a simple handshake between the ISA bus and the NIC core through a D-type data latch (internal to the AT/LANTIC). For a remote DMA read, the AT/LANTIC will read a byte/word of data from the local memory and write it to the data latch. At the same time an I/O read cycle is executed by the host and the data is read from the latch. The handshake logic will wait state the ISA bus if the data is not ready when the read strobe becomes active. In a like manner, the AT/LANTIC will read data from the latch and write it to the local RAM during a remote DMA write.

As in Shared RAM mode, the AT/LANTIC's registers are accessible in I/O space. There is the addition of a hardware RESET port which can be driven by reading and writing to I/O location BASE + 18H, where BASE is the I/O base address chosen in Configuration Register A. The registers, data latches and reset port occupy 16 bytes of I/O space. Unlike Shared RAM mode, the Ethernet Address PROM is not located in I/O space. It is accessible only by the AT/LANTIC, so a remote DMA read is used to obtain its contents. In both I/O mode and Shared RAM mode, the Ethernet Address PROM is not a physical device, but a dynamic copy of the Ethernet address and other information (from the E<sup>2</sup>PROM) held in the AT/LANTIC.

### 5.3 Transmit

The DP83905EB-AT can transmit and receive Ethernet packets on the three basic types of media available today. A twisted pair transceiver internal to the AT/LANTIC allows easy connectivity to a 10BASE-T network. The transceiver

provides link integrity checking, polarity detection and correction and pre-emphasis of the transmit output. The AT/LANTIC can also be configured to use its AUI interface for connection to a MAU (10BASE2/5/F/T) or on board 10BASE2 transceiver.

To transmit a packet, there are three basic steps that need to be taken. Please refer to *Figure 1* of this Application Note. First, the packet that is to be transmitted is loaded into the AT/LANTIC's buffer RAM by the host. The packet must consist of a destination address, a source address, length and data. If the data is less than 48 bytes, it must be padded with arbitrary data so that the entire packet is 64 bytes as required by the IEEE 802.3 standard. (This is not a requirement of the AT/LANTIC.) Next, once the packet is in RAM, the host programs the local DMA registers with the location and length of the packet to be transmitted. Lastly, the host issues the transmit command to the AT/LANTIC and the AT/LANTIC will start transmitting the packet.

The first activity the AT/LANTIC does is a prefetch to load the FIFO with data. Next, the NIC core of the AT/LANTIC will start sending preamble in NRZ format to the ENDEC. The ENDEC, in turn, encodes the data with the transmit clock in Manchester format and drives a differential transmit pair to either the internal twisted pair transceiver or to the AUI port of the AT/LANTIC. The twisted pair transceiver then drives the TXO outputs while pre-emphasizing the signal to eliminate inter-symbol jitter. If 10BASE2 is being used, the DP8392CV receives the differential transmit data from the AT/LANTIC and drives the coax line with a single-ended signal. Both transceivers will monitor the network for collisions.

In the event that there is a collision, the transceiver will drive a 10 MHz signal on the Collision Detect differential pair. The ENDEC will then drive a collision signal to the NIC core so that the Ethernet MAC control section of the NIC will implement the collision backoff algorithm and attempt re-transmission.

After the preamble and start-of-frame delimiter have been transmitted, the AT/LANTIC will serialize bytes from the FIFO and transmit them in the manner described above. When the FIFO empties to a pre-set threshold, the AT/LANTIC will fetch more data from the local RAM using its local DMA function. As the data is being transmitted, a Cyclic Redundancy Check (CRC) is continuously being calculated. After all the data has been transmitted, the 4 byte CRC value is transmitted. This allows receiving stations to check the received packet for data integrity.

### 5.4 Receive

Receiving a packet is almost the exact opposite of transmitting a packet. The functional blocks discussed above remain the same, however there are some differences which will be discussed. The transceiver will sit idle until a valid signal is received on the media, indicating the start of a packet. This causes the transceiver's RX squelch to turn off. Both the twisted pair transceiver (internal to the AT/LANTIC) and the DP8392CV CTI have squelch circuitry which filter out noise on the network, however, they are implemented differently. The TPI transceiver checks an incoming signal for both amplitude and frequency (sometimes referred to as smart squelch).

The CTI only checks for valid amplitude. Once the CTI has turned its RX squelch off it starts driving the receive differential pair of the AUI. The ENDEC (internal to the AT/LANTIC) recovers a 10 MHz clock and receive data in NRZ format from the receive pair with its analog PLL. The NRZ data is clocked into the NIC core, de-serialized, and loaded into the FIFO. When the number of bytes in the FIFO reaches the programmed threshold, the AT/LANTIC requests the local bus and writes the received data into the receive buffer memory using its local DMA. After the packet is received, the AT/LANTIC writes four bytes of status information into the buffer and interrupts the host to inform it that the packet can be removed by software.

## 6.0 CONCLUSION

The DP83905EB-AT Demonstration board provides system developers with an easy to understand and use example of how to design an industry standard Ethernet adapter card. The DP83905EB-AT design can readily be adapted for PC® motherboard designs too.

By using the DP83905 AT/LANTIC, jumperless Ethernet boards can be made extremely cost effective and simple to design and use.

### Parts List for DP83905EB-AT

#### Capacitors

C1	27 pF	10% Ceramic, SMT 0805
C2	27 pF	10% Ceramic, SMT 0805
C3-C4	0.01 $\mu$ F/50V	20% Ceramic, SMT 1206
C5	22 $\mu$ F/16V	20% Tantalum, SMT 7343
C6	0.01 $\mu$ F/50V	20% Ceramic, SMT 1206
C7	0.01 $\mu$ F/1 kV	20% Ceramic Disk, TH
C8-C9	0.01 $\mu$ F/50V	20% Ceramic, SMT 1206
C10	10 $\mu$ F/16V	10% Ceramic, SMT 7343
C11-C20	0.01 $\mu$ F/50V	20% Ceramic, SMT 1206
C21-C22	22 $\mu$ F/16V	20% Tantalum, SMT 7343
C23-C27	0.01 $\mu$ F/50V	20% Ceramic, SMT 1206
C28-C31	0.01 $\mu$ F/50V	20% Ceramic, SMT 1206
C42, C44	0.01 $\mu$ F/50V	20% Ceramic, SMT 1206
C43, C45	0.001 $\mu$ F/50V	20% Ceramic, SMT 1206
C46-C47	33 pF/50V	10% Ceramic, SMT 0805
SP1	0.75 pF/1 kV	Spark Gap, TH

#### Resistors

R2-R5	270 $\Omega$	5%, $\frac{1}{8}$ W, SMT 1206
R6-R9	39.2 $\Omega$	1%, $\frac{1}{8}$ W, SMT 1206
R10-R13	1.5 k $\Omega$	5%, $\frac{1}{8}$ W, SMT 1206
R14	1 k $\Omega$	1%, $\frac{1}{8}$ W, SMT 1206
R15	150 $\Omega$	1%, $\frac{1}{8}$ W, SMT 1206
R17	1 M $\Omega$	5%, $\frac{1}{2}$ W, TH
R18	10 k $\Omega$	1%, $\frac{1}{8}$ W, SMT 1206
R19	274 $\Omega$	1%, $\frac{1}{8}$ W, SMT 1206
R20	66.5 $\Omega$	1%, $\frac{1}{8}$ W, SMT 1206
R21	66.5 $\Omega$	1%, $\frac{1}{8}$ W, SMT 1206
R22	274 $\Omega$	1%, $\frac{1}{8}$ W, SMT 1206
R23	806 $\Omega$	1%, $\frac{1}{8}$ W, SMT 1206
R24-R25	49.9 $\Omega$	1%, $\frac{1}{8}$ W, SMT 1206
R26	22 $\Omega$	1%, $\frac{1}{8}$ W, SMT 1206
R39-R41	270 $\Omega$	5%, $\frac{1}{8}$ W, SMT 1206
R42	1 k $\Omega$	1%, $\frac{1}{8}$ W, SMT 1206

## Parts List for DP83905EB-AT (Continued)

**Integrated Circuits**

U1	AT/LANTIC	DP83905		
U4	EEPROM	NM93C06		
U5	EPROM	NM27C256		Do not populate
U6	FILTER	FL1012	VALOR	
U7	CTI	DP8392CV		
U9	8k x 8 SRAM	NMS64X8M70		
U10	8k x 8 SRAM	NMS64X8M70		
U13	GAL16V8	16V8-25LVC		

**Connectors**

J3	15-Pin D Connector, Female (AMP # 745782-4)			AMP
J4	BNC Connector, Female (AMP # 227161-7)			AMP
J5	RJ45 Modular Phone Jack, 8-Pin (AMP # 555164-1)			AMP

**Magnetics**

U8	DC-DC CONVERTER	PM6077	VALOR	
Y1	PULSE TRANSFORMER	23Z91SM	FIL-MAG	
Y2	PULSE TRANSFORMER	23Z91SM	FIL-MAG	

**Socket**

S1	28-Pin Dual-in-Line Socket for EPROM, U5			
----	--	--	--	--

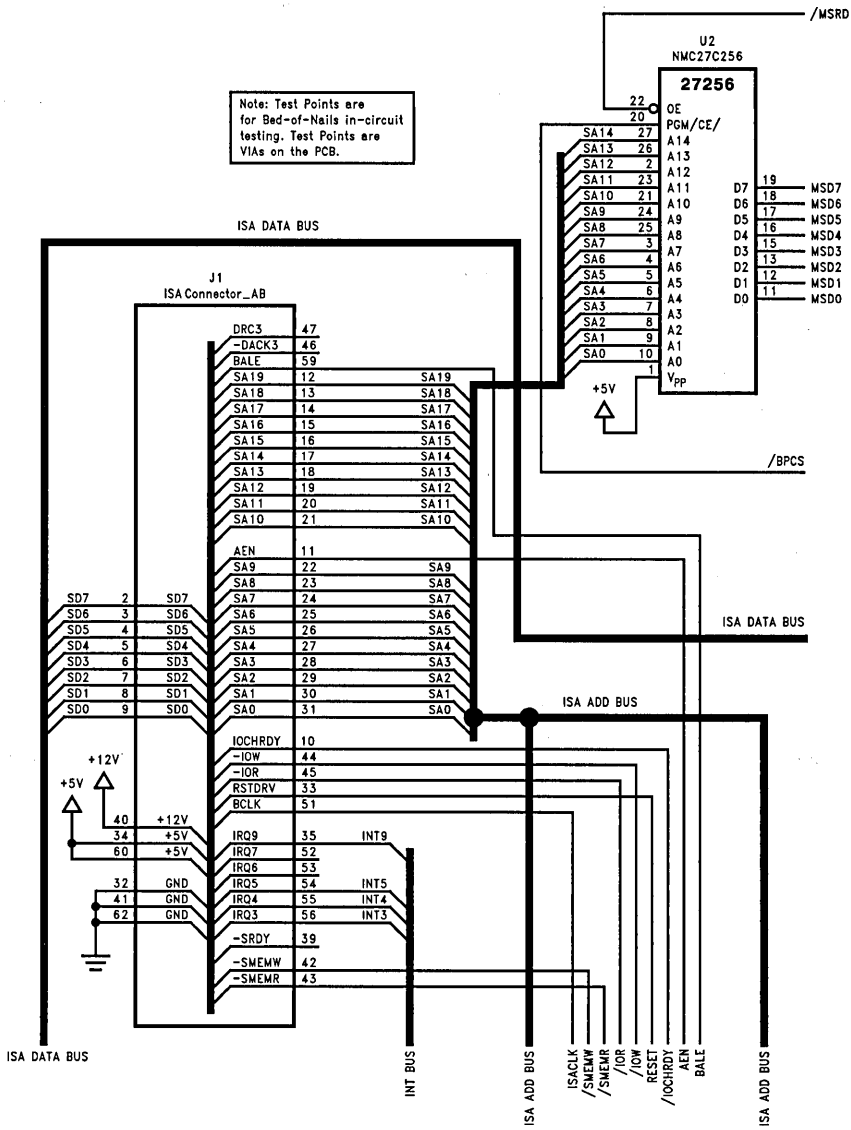
**Diodes**

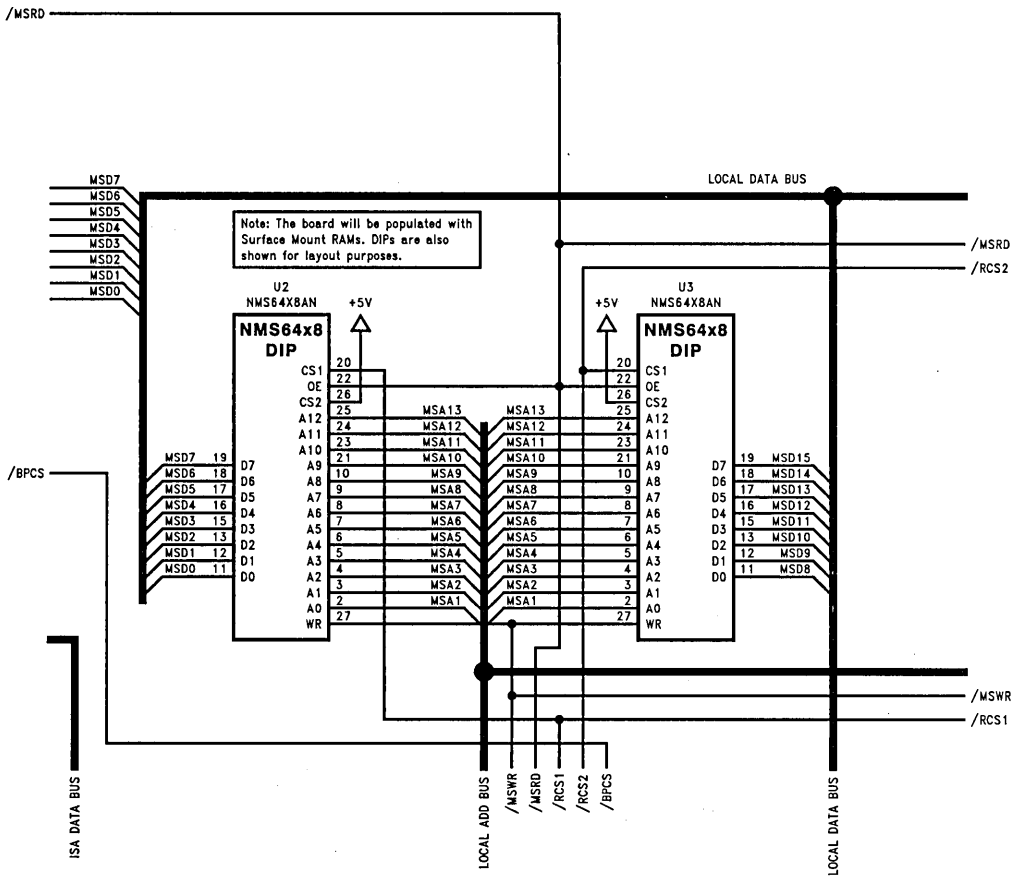
D1	MMBD1201			
D2	3mm Green LED			
D3	3mm Orange LED			
D6	3mm Green LED			
D7	3mm Hi-Efficiency Red LED			
D8	3mm Yellow LED			

**Clocks**

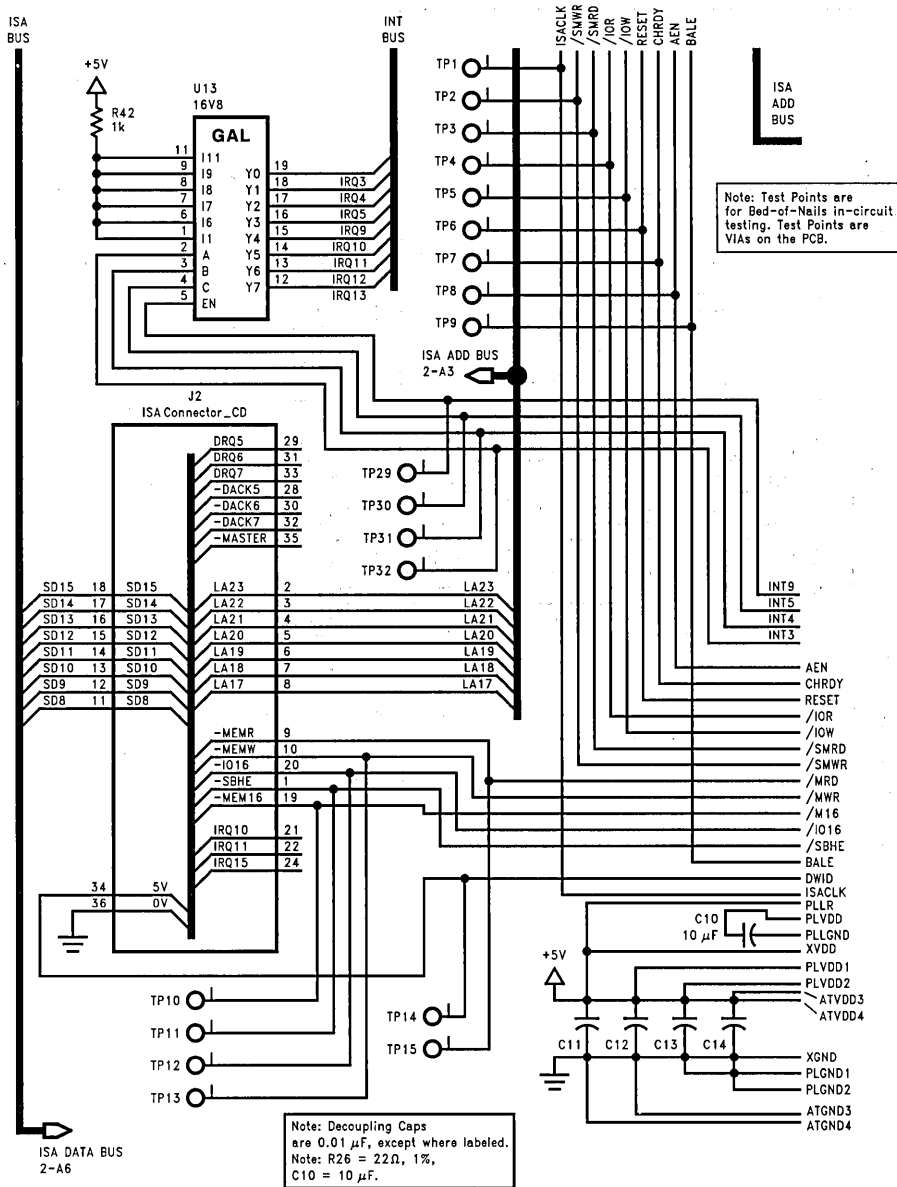
X1	HC49S, 20 MHz Crystal (low profile) (Refer to AT/LANTIC datasheet for crystal specifications).			
X2	20 MHz Oscillator, 45-55 Duty cycle, 0.001% Tolerance Oscillator module should be raised off of board when mounting.			Do not populate

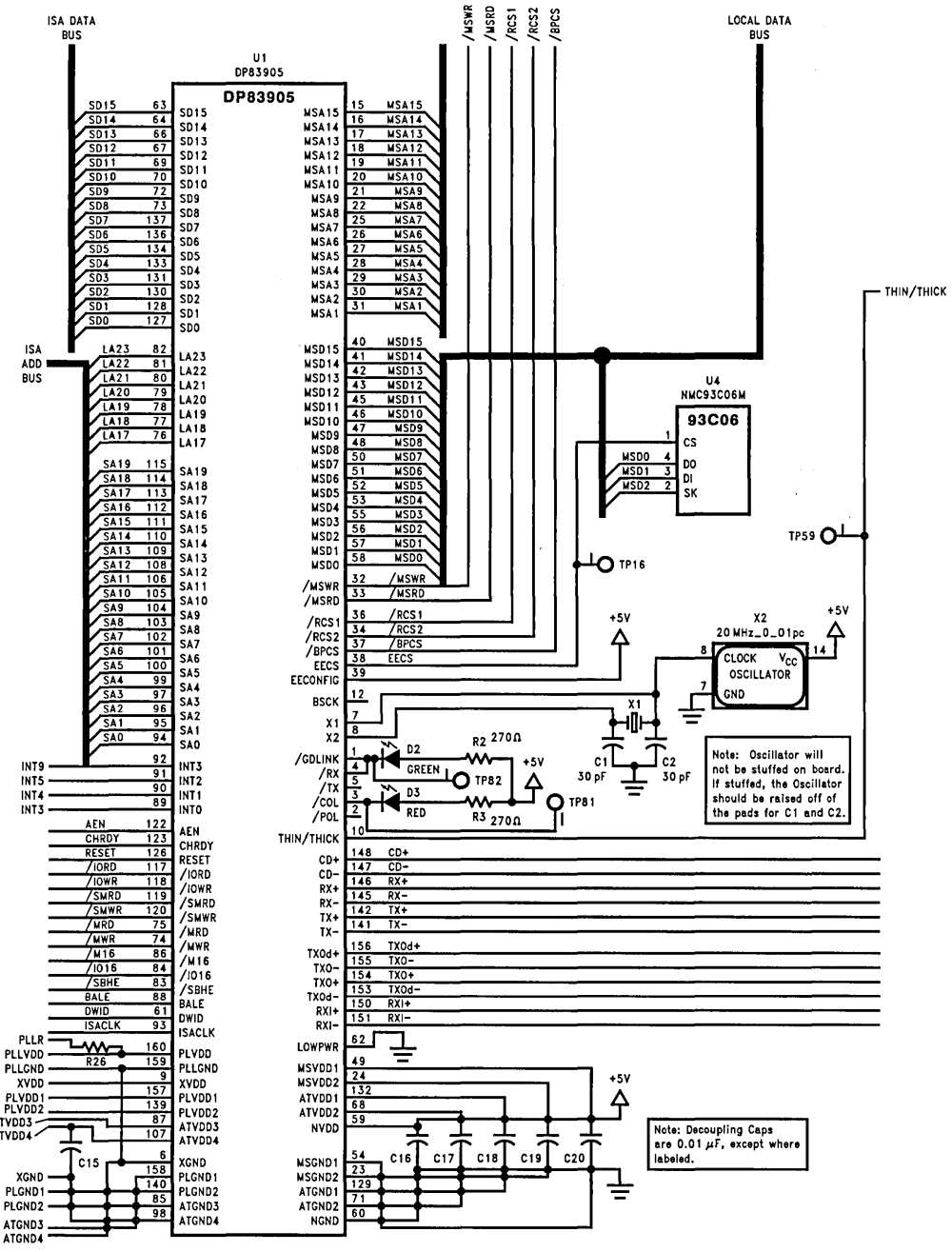
Note: Test Points are for Bed-of-Nails in-circuit testing. Test Points are VIAs on the PCB.





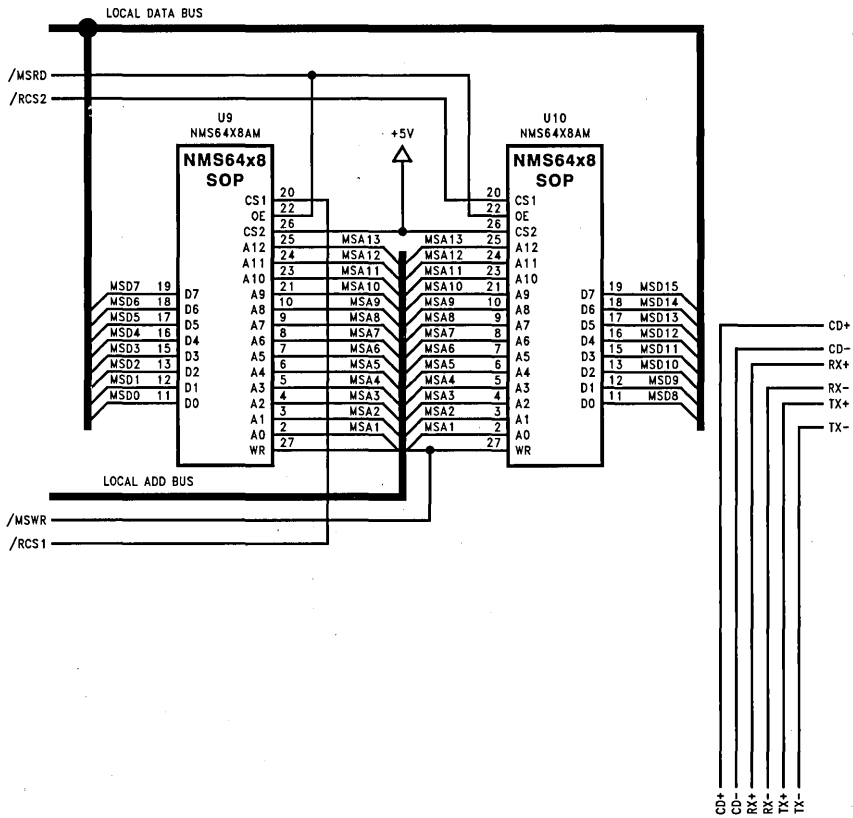
TL/F/11786-5





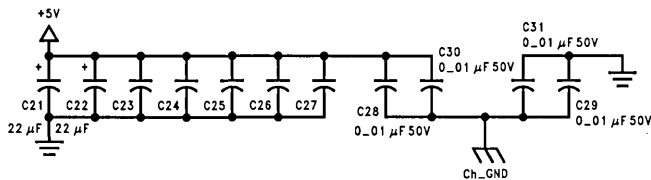
TL/F/11786-7





TL/F/11786-8



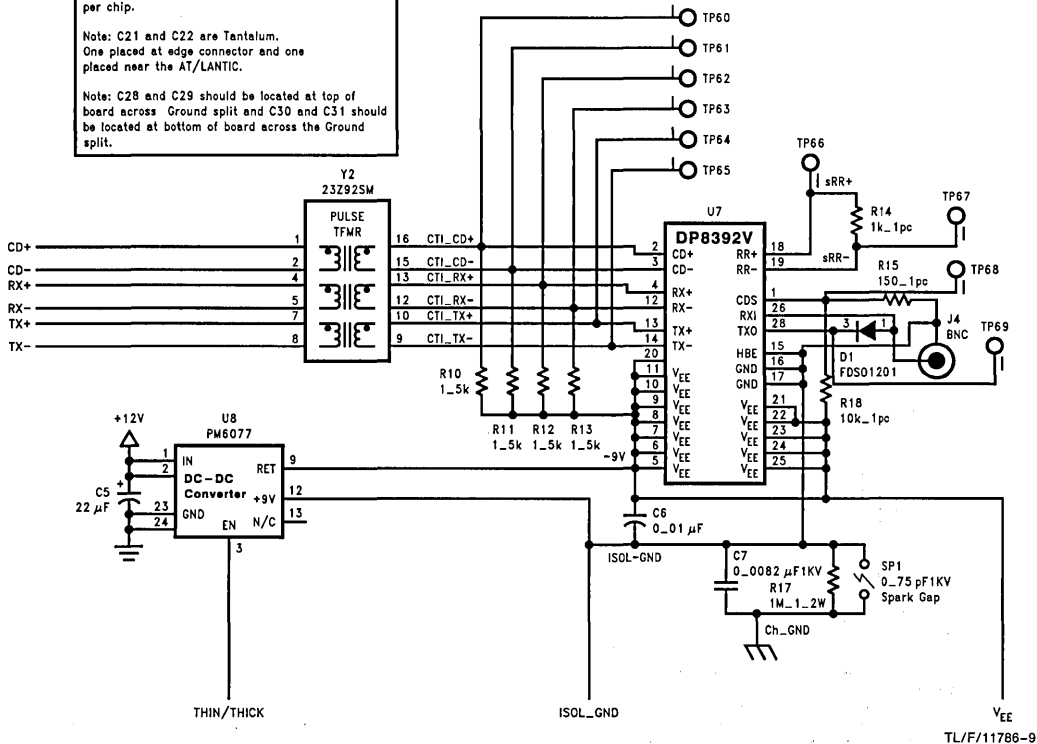


Note: Decoupling Caps are 0.01  $\mu\text{F}$  except where labeled.

Note: Locate one 0.01  $\mu\text{F}$  Cap per chip.

Note: C21 and C22 are Tantalum. One placed at edge connector and one placed near the AT/LANTIC.

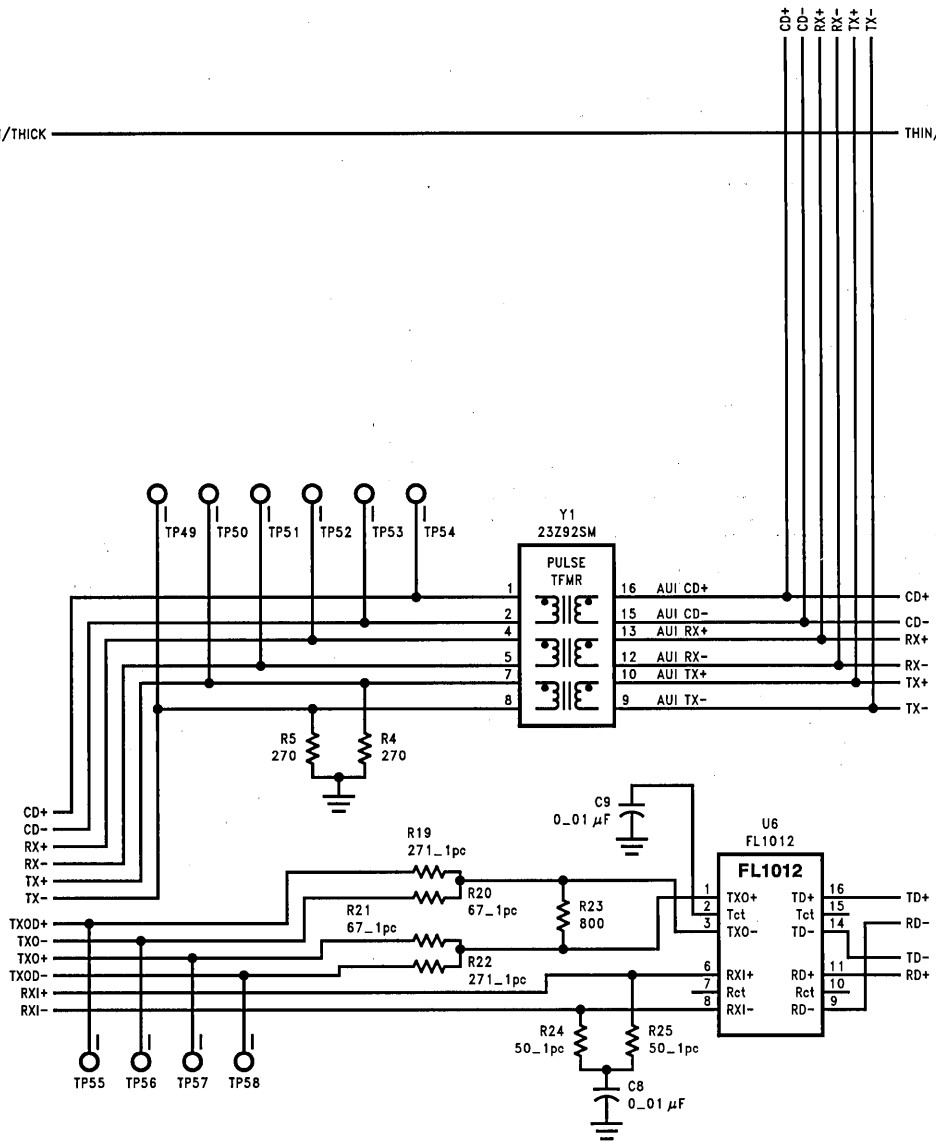
Note: C28 and C29 should be located at top of board across Ground split and C30 and C31 should be located at bottom of board across the Ground split.



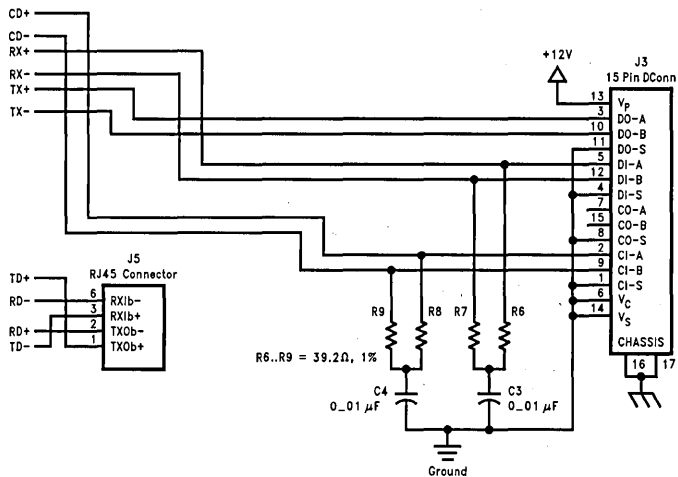
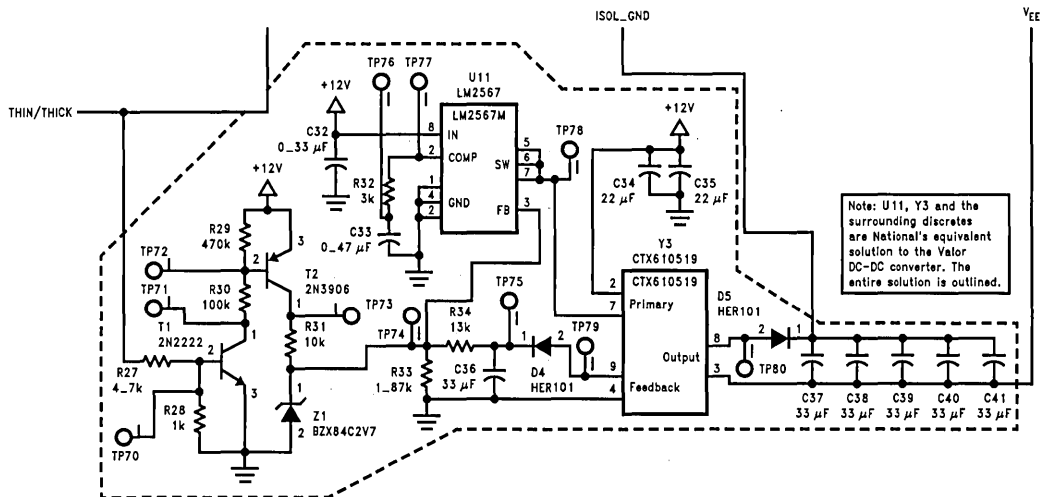
TL/F/11786-9

THIN/THICK

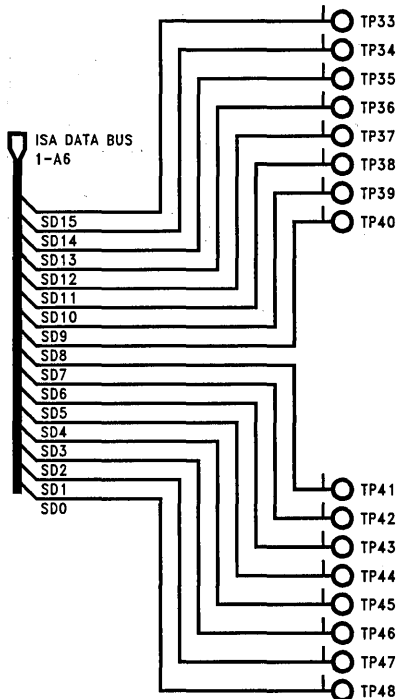
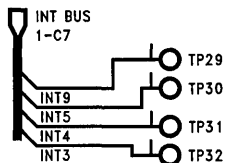
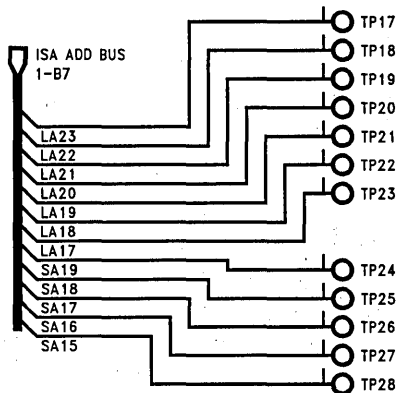
THIN/THICK



TL/F/11786-10



TL/F/11786-11



NOTE: These Test Points are for In-Circuit bed-of-nails testing. They will be VIAs on the PCB.

TL/F/11786-12

# The Operation of the FIFO in the DP8390, DP83901, DP83902 and DP83905

National Semiconductor  
Application Note 886  
Bonnie Wilson  
Bill Lee



## 1.0 INTRODUCTION

To accommodate the different rates at which data comes from (or goes to) the network and goes to (or comes from) the system memory, the NIC contains a 16-byte FIFO for buffering data between the bus and the media. The FIFO threshold is programmable, allowing filling (or emptying) the FIFO with different burst lengths. When the FIFO has filled to its programmed threshold, the local DMA channel transfers these bytes or words into local memory. It is crucial that the local DMA is given access to the bus within a minimum bus latency time, otherwise a FIFO underrun (or overrun) occurs. During transmission the DMA writes data into the FIFO and the Transmit Serializer reads data from the FIFO and transmits it. During reception the Receive Deserializer writes data into the FIFO and the DMA reads data from the FIFO.

## 2.0 FIFO THRESHOLD

The DMA transfers between the FIFO and memory occur in bursts beginning when the FIFO threshold is reached. The threshold takes on different meanings during transmission and reception. During reception the FIFO threshold refers to the number of bytes in the FIFO. During transmission the FIFO threshold refers to the number of empty bytes in the FIFO: the size of the FIFO (16) - # bytes in FIFO. Bits FT0 and FT1 in the Data Configuration Register set the FIFO threshold to 2 bytes, 4 bytes, 8 bytes, or 12 bytes (1 word, 2 words, 4 words, or 6 words).

The threshold for the first burst is different than subsequent thresholds as discussed in detail in Sections 3.0 and 4.0. The values in Tables I and II are derived from the timing diagrams in Section 6.0. The first threshold refers to the state of the FIFO at point B in the timing diagrams, and the threshold refers to the state of the FIFO at point D. The discussion below refers to the threshold, not the first threshold.

The FIFO logic operates differently in reception and transmission. During reception in byte mode, a threshold is indicated when approximately the  $n + 14$ th bit has entered the FIFO; thus, with an 8-byte threshold, the NIC issues Bus Request (BREQ) when the FIFO contains 9 bytes and 6 bits. For reception in word mode, BREQ is generated when approximately  $n + 22$  bits have entered the FIFO; thus with a 2-word threshold, BREQ is issued when the 54th bit has entered the FIFO. Refer to Table I for the exact receive thresholds for each case.

During transmission in byte mode, a threshold is indicated when approximately the  $n + 12$ th bit has entered the FIFO; thus with an 8-byte threshold, the NIC issues BREQ when the FIFO contains 9 bytes and 4 bits. For transmission in word mode, BREQ is generated when approximately  $n + 29$  bits have entered the FIFO. Thus, with a 4-word threshold (equivalent to an 8-byte threshold), BREQ is issued when the 96th bit has entered the FIFO. Refer to Table II for the exact transmit thresholds for each case.

TABLE I. Receive Packet Thresholds

Receive Packet Cases	First Threshold	Threshold
Word Mode, 1 Word Threshold	4 Words, 11 bits	2 Words, 5 bits
Word Mode, 2 Word Threshold	4 Words, 10 bits	3 Words, 6 bits
Word Mode, 4 Word Threshold	5 Words, 7 bits	5 Words, 5 bits
Word Mode, 6 Word Threshold	6 Words, 8 bits	6 Words, 6 bits
Byte Mode, 2 Byte Threshold	9 Bytes, 2 bits	3 Bytes, 4 bits
Byte Mode, 4 Byte Threshold # 1	9 Bytes, 2 bits	5 Bytes, 6 bits
Byte Mode, 4 Byte Threshold # 2	9 Bytes, 6 bits	5 Bytes, 6 bits
Byte Mode, 8 Byte Threshold	10 Bytes	9 Bytes, 6 bits
Byte Mode, 12 Byte Threshold	13 Bytes, 7 bits	13 Bytes, 5 bits

TABLE II. Transmit Packet Thresholds

Transmit Packet Cases	First Threshold	Threshold
Word Mode, 1 Word Threshold	6 Words	3 Words, 12 bits
Word Mode, 2 Word Threshold	4 Words, 3 bits	3 Words, 13 bits
Word Mode, 4 Word Threshold	5 Words, 13 bits	5 Words, 13 bits
Word Mode, 6 Word Threshold	7 Words, 13 bits	7 Words, 13 bits
Byte Mode, 2 Byte Threshold	12 Words, 1 bit	(See Timing Diagram, Figure 14)
Byte Mode, 4 Byte Threshold	12 Bytes, 1 bit	5 Bytes, 5 bits
Byte Mode, 8 Byte Threshold	9 Bytes, 6 bits	9 Bytes, 4 bits
Byte Mode, 12 Byte Threshold	13 Bytes, 6 bits	13 Bytes, 4 bits

### 3.0 FIFO OPERATION DURING RECEIVE

At the beginning of reception, the NIC stores the entire Address field of each incoming packet in the FIFO to determine whether the packet matches its Physical Address Registers or maps to one of its Multicast Registers. Therefore, the first local DMA transfer does not occur until after 8 bytes (4 words) have accumulated in the FIFO, regardless of the value of the threshold. This affects the bus latencies at 2 byte, 4 byte, 1 word, and 2 word thresholds during the first receive BREQ. Thus the threshold for the first burst that is loaded into memory differs from the remaining threshold values. Refer to Table I for the exact threshold values.

### 4.0 FIFO OPERATION DURING TRANSMIT

Before transmitting, the NIC performs a prefetch from memory to load the FIFO. The number of bytes prefetched is the programmed FIFO threshold, except for 1 byte, 1 word, and 2 word thresholds which prefetch 4 bytes, 2 words, and 4 words respectively. The next BREQ is not issued until after the NIC actually begins transmitting data, i.e., after Preamble and SFD. The threshold for the first burst that is loaded from memory following the prefetched data often differs from the remaining threshold values. Refer to Table II for the exact threshold values.

### 5.0 FIFO UNDERRUNS AND OVERRUNS

To assure that there is no overwriting of data, the FIFO logic flags an overrun if it becomes full before bus acknowledge is returned. To assure that there is no lost data, the FIFO flags an underrun if it becomes empty before bus acknowledge is returned. There are two causes which produce overruns and underruns:

1. The bus latency is so long that the FIFO has filled (or emptied) before the local DMA has serviced the FIFO.
2. The bus latency or bus data rate has slowed the throughput of the local DMA to a point where it is slower than the network data rate (10 Mb/s). This second condition is also dependent upon DMA clock and data width (byte wide or word wide).

The worst case condition ultimately limits the overall bus latency that the NIC can tolerate.

### 6.0 TIMING DIAGRAMS

The following pages contain detailed timing diagrams and descriptions of every possible receive and transmit mode transfer. The descriptions are of 2, 4, 8, and 12 byte transfers (on an 8-bit Novell board) and 1, 2, 4, and 6 word transfers (on a 16-bit Novell board). For each transfer, the bus clock runs at 20.0 MHz. Tables III and IV summarize the information found in the receive and transmit transfer diagrams respectively.

**TABLE III. Receive Packet Transfers**

Receive Packet Cases	After SFD to BREQ Asserted (Point A to B)	BREQ Asserted to MWR Deasserted (Point B to C)	MWR Deasserted to BREQ Asserted (Point C to D)	BREQ Asserted to MWR Deasserted (Point D to E)
Word Mode 1 Word Threshold	75 ± 4 bits Shifted In	7 Words Removed 52 bits Shifted In	22 bits Shifted In	2 Words Removed 12 bits Shifted In
Word Mode 2 Word Threshold	74 ± 2 bits Shifted In	4 Words Removed 16 bits Shifted In	28 bits Shifted In	2 Words Removed 6 bits Shifted In
Word Mode 4 Word Threshold	87 ± 2 bits Shifted In	4 Words Removed 9 bits Shifted In	53 bits Shifted In	4 Words Removed 9 bits Shifted In
Word Mode 6 Word Threshold	104 ± 2 bits Shifted In	6 Words Removed 14 bits Shifted In	80 bits Shifted In	6 Words Removed 14 bits Shifted In
Byte Mode 2 Byte Threshold	74 ± 2 bits Shifted In	18 Bytes Removed 86 bits Shifted In	12 bits Shifted In	2 Bytes Removed 6 bits Shifted In
Byte Mode 4 Byte Threshold # 1	74 ± 1 bit Shifted In	8 Bytes Removed 23 bits Shifted In	21 bits Shifted In	4 Bytes Removed 10 bits Shifted In
Byte Mode 4 Byte Threshold # 2	78 ± 1 bit Shifted In	12 Bytes Removed 38 bits Shifted In	26 bits Shifted In	4 Bytes Removed 10 bits Shifted In
Byte Mode 8 Byte Threshold	80 ± 2 bits Shifted In	8 Bytes Removed 18 bits Shifted In	44 bits Shifted In	8 Bytes Removed 18 bits Shifted In
Byte Mode 12 Byte Threshold	111 ± 2 bits Shifted In	12 Bytes Removed 26 bits Shifted In	68 bits Shifted In	12 Bytes Removed 26 bits Shifted In

**TABLE IV. Transmit Packet Transfers**

Transmit Packet Cases	Initial Loading to BREQ Asserted (Point A to B)	BREQ Asserted to MRD Deasserted (Point B to C)	MRD Deasserted to BREQ Asserted (Point C to D)	BREQ Asserted to MRD Deasserted (Point D to E)
Word Mode 1 Word Threshold	2 Words Loaded 0 bits Shifted Out	7 Words Loaded 52 bits Shifted Out	24 bits Shifted Out	2 Words Loaded 12 bits Shifted Out
Word Mode 2 Word Threshold	4 Words Loaded 3 bits Shifted Out	2 Words Loaded 6 bits Shifted Out	20 bits Shifted Out	2 Words Loaded 6 bits Shifted Out
Word Mode 4 Word Threshold	4 Words Loaded 29 bits Shifted Out	4 Words Loaded 14 bits Shifted Out	54 bits Shifted Out	4 Words Loaded 9 bits Shifted in
Word Mode 6 Word Threshold	6 Words Loaded 93 bits Shifted Out	6 Words Loaded 14 bits Shifted in	82 bits Shifted Out	6 Words Loaded 14 bits Shifted in
Byte Mode 2 Byte Threshold	4 Bytes Loaded 1 bit Shifted Out	(Refer to Timing Diagram <i>Figure 14</i> )	(Refer to Timing Diagram <i>Figure 14</i> )	(Refer to Timing Diagram <i>Figure 14</i> )
Byte Mode 4 Byte Threshold	4 Bytes Loaded 1 bit Shifted Out	15 Bytes Loaded 49 bits Shifted Out	19 bits Shifted Out	4 Bytes Loaded 10 bits Shifted Out
Byte Mode 8 Byte Threshold	8 Bytes Loaded 22 bits Shifted Out	8 Bytes Loaded 18 bits Shifted Out	44 bits Shifted Out	8 Bytes Loaded 18 bits Shifted Out
Byte Mode 12 Byte Threshold	12 Bytes Loaded 78 bits Shifted Out	12 Bytes Loaded 26 bits Shifted Out	68 bits Shifted Out	12 Bytes Loaded 26 bits Shifted Out

RECEIVE PACKET

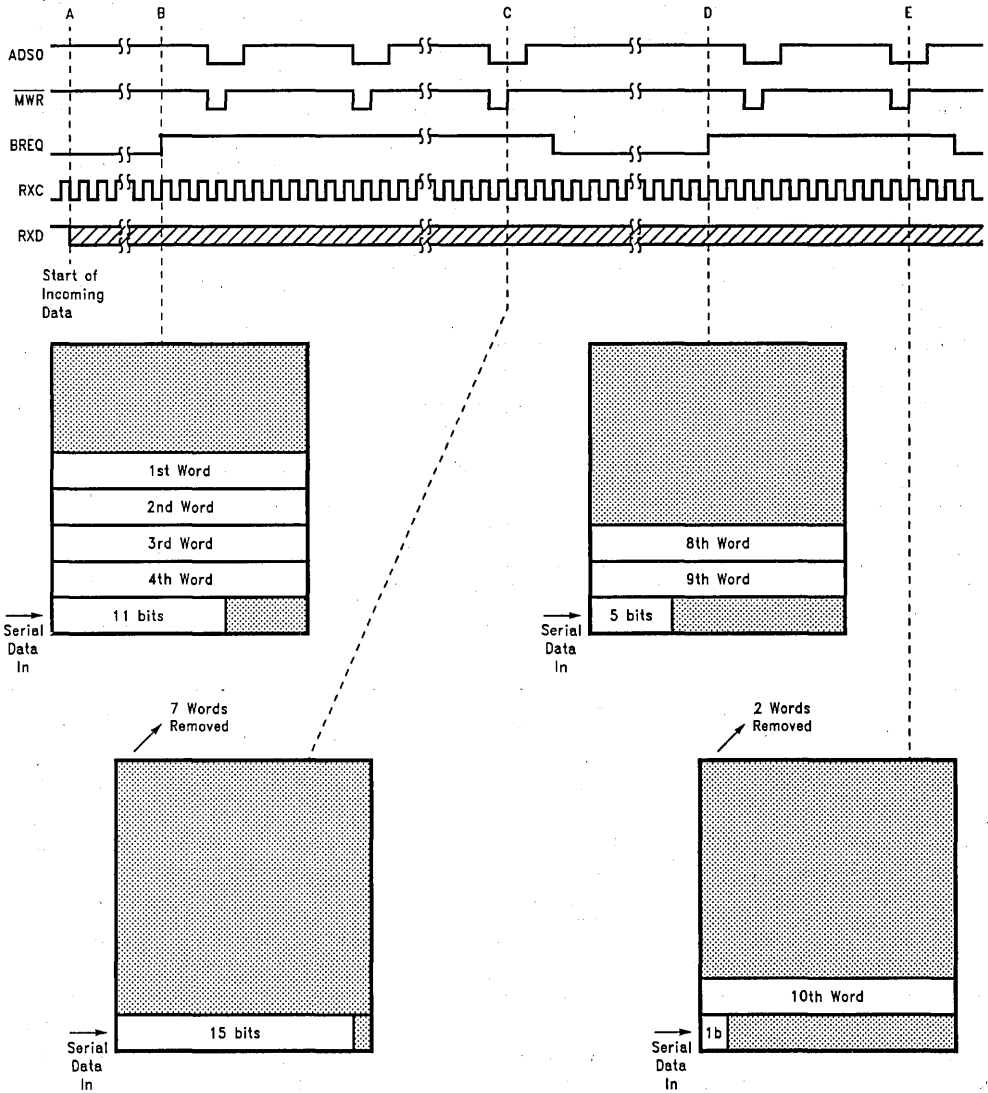


FIGURE 1. Receive Packet—Word Mode, 1 Word Threshold

TL/F/11819-1

After the preamble and the Start of Frame Delimiter (SFD) (Point A), the FIFO shifts in  $75 \pm 4$  bits of data (depending on the location of the SFD with respect to  $t_1$  of the DMA sequence) before BREQ is asserted (Point B). The FIFO transfers 7 words into memory while shifting in 52 bits of data until MWR is deasserted (Point C). BREQ is asserted

when there are 2 words and 5 bits in the FIFO (Point D). Two words are removed from the FIFO and stored in memory, while 12 bits are shifted into the FIFO until MWR is deasserted (Point E). D and E are repeated until the packet is complete.



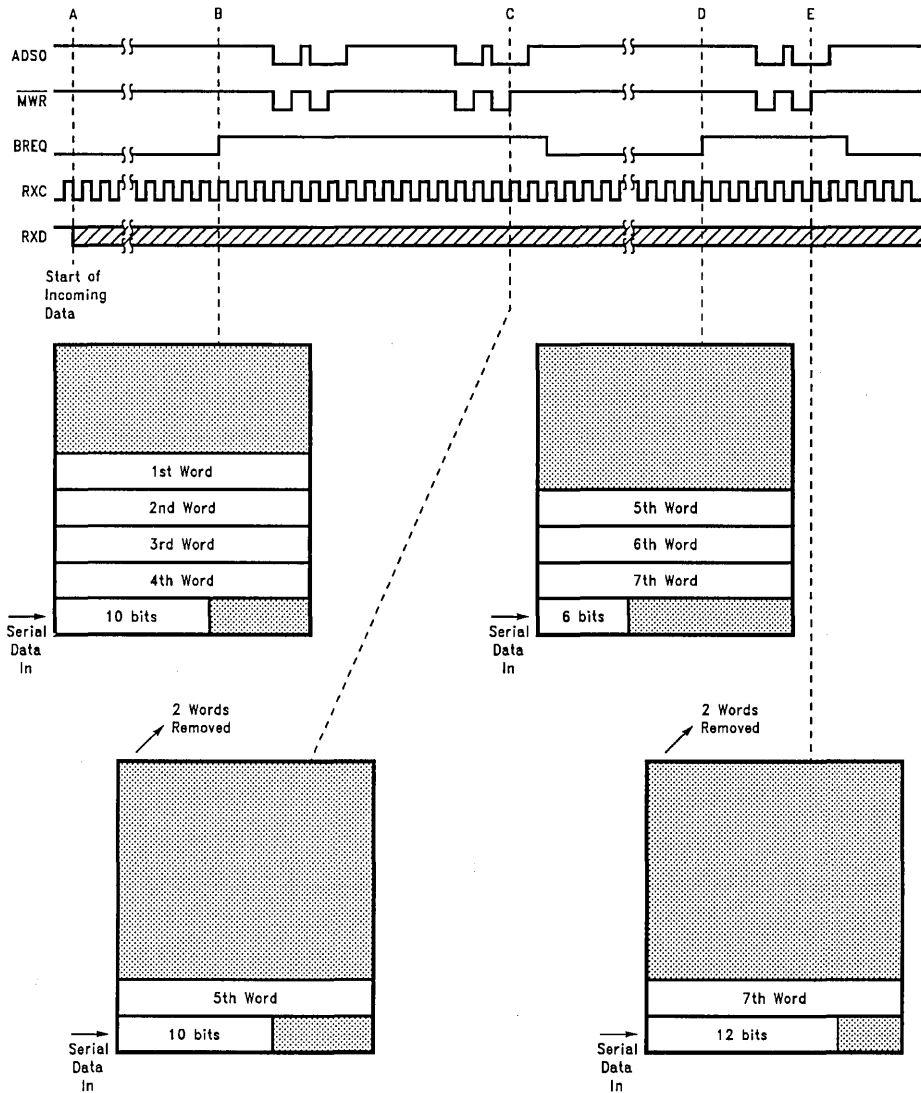
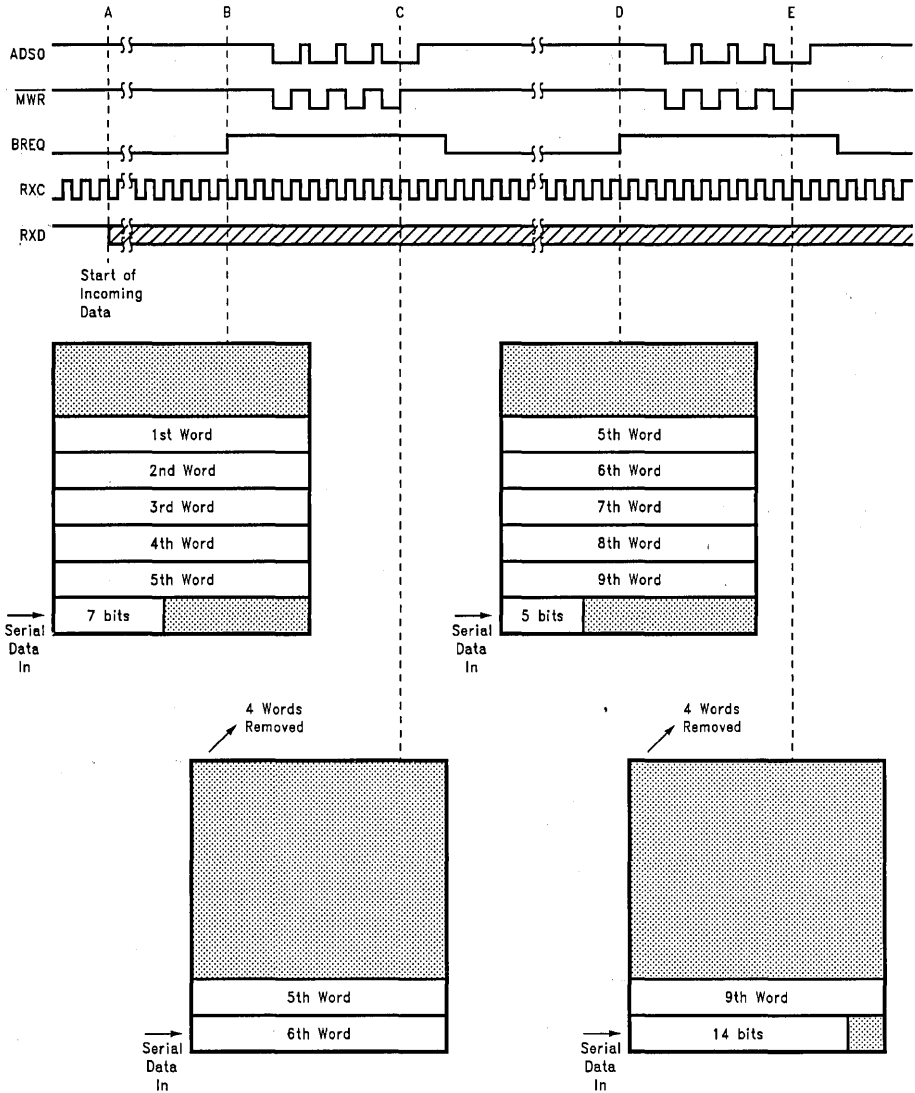


FIGURE 2. Receive Packet—Word Mode, 2 Word Threshold

TL/F/11819-2

After the preamble and the SFD (Point A), the FIFO shifts in  $74 \pm 2$  bits of data (depending on the location of the SFD with respect to  $t_1$  of the DMA sequence) before BREQ is asserted (Point B). The FIFO transfers 2 bursts of 2 words each into memory while shifting in 16 bits of data until

$\overline{MWR}$  is deasserted (Point C). BREQ is asserted when there are 3 words and 6 bits in the FIFO (Point D). Two words are removed from the FIFO and stored in memory, while 6 bits are shifted into the FIFO until  $\overline{MWR}$  is deasserted (Point E). D and E are repeated until the packet is complete.



**FIGURE 3. Receive Packet—Word Mode, 4 Word Threshold**

TL/F/11819-3

After the preamble and the SFD (Point A), the FIFO shifts in  $87 \pm 2$  bits of data (depending on the location of the SFD with respect to  $t_1$  of the DMA sequence) before BREQ is asserted (Point B). The FIFO transfers 1 burst of 4 words into memory while shifting in 9 bits of data until MWR is

deasserted (Point C). BREQ is asserted when there are 5 words and 5 bits in the FIFO (Point D). Again, 4 words are removed from the FIFO and stored in memory, while 9 bits are shifted into the FIFO until MWR is deasserted (Point E). D and E are repeated until the packet is complete.

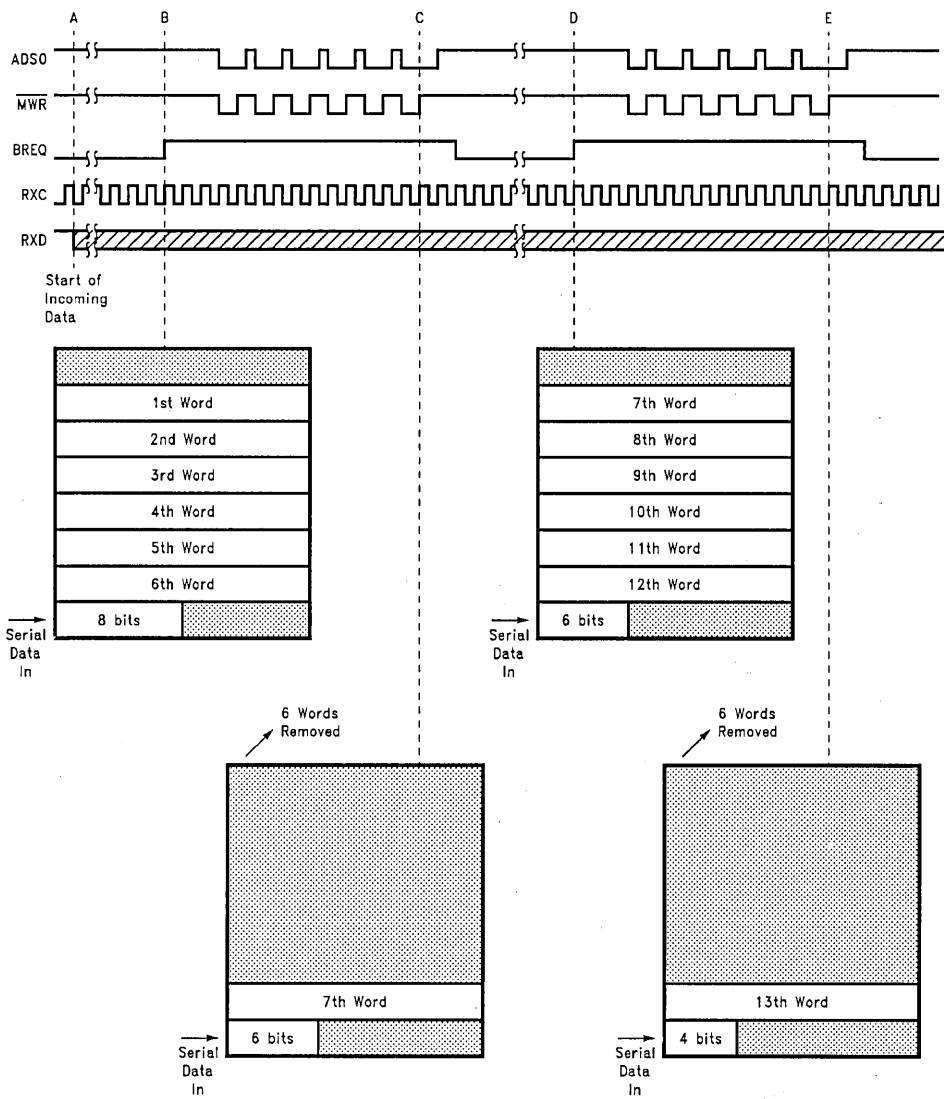
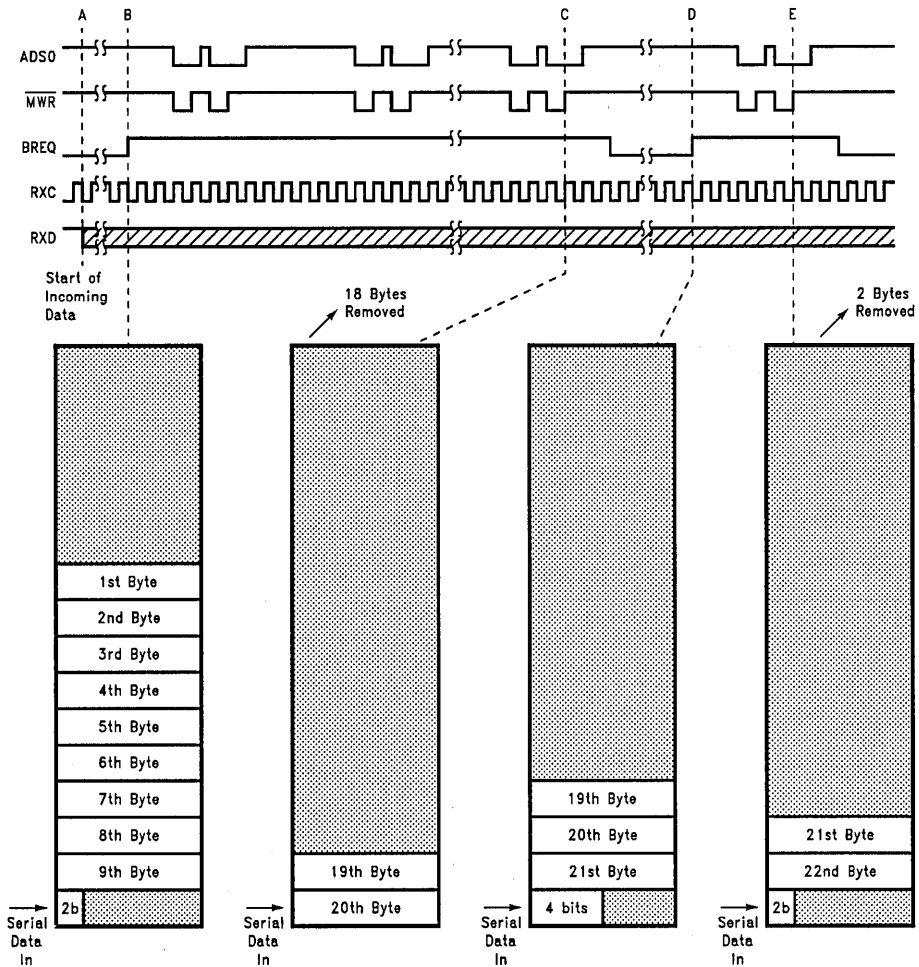


FIGURE 4. Receive Packet—Word Mode, 6 Word Threshold

TL/F/11819-4

After the preamble and the SFD (Point A), the FIFO shifts in  $104 \pm 2$  bits of data (depending on the location of the SFD with respect to  $t_1$  of the DMA sequence) before BREQ is asserted (Point B). The FIFO transfers 1 burst of 6 words into memory while shifting in 14 bits of data until  $\overline{MWR}$  is

deasserted (Point C). BREQ is asserted when there are 6 words and 6 bits in the FIFO (Point D). Again, 6 words are removed from the FIFO and stored in memory, while 14 bits are shifted into the FIFO until  $\overline{MWR}$  is deasserted (Point E). D and E are repeated until the packet is complete.

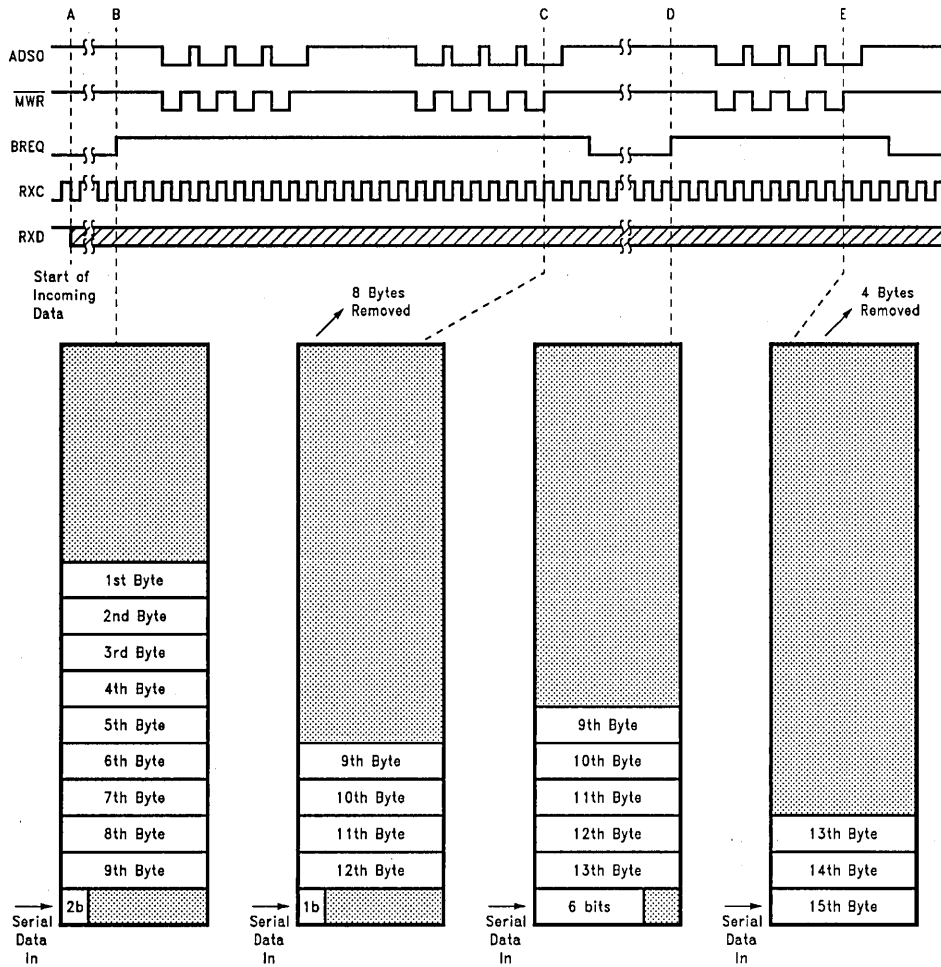


**FIGURE 5. Receive Packet—Byte Mode, 2 Byte Threshold**

TL/F/11819-5

After the preamble and the SFD (Point A), the FIFO shifts in  $74 \pm 2$  bits of data (depending on the location of the SFD with respect to  $t_1$  of the DMA sequence) before BREQ is asserted (Point B). The FIFO transfers 9 bursts of 2 bytes each into memory while shifting in 86 bits of data until  $\overline{MWR}$

is deasserted (Point C). BREQ is asserted when there are 12 bytes and 4 bits in the FIFO (Point D). Two bytes are removed from the FIFO and stored in memory, while 6 bits are shifted into the FIFO until  $\overline{MWR}$  is deasserted (Point E). D and E are repeated until the packet is complete.

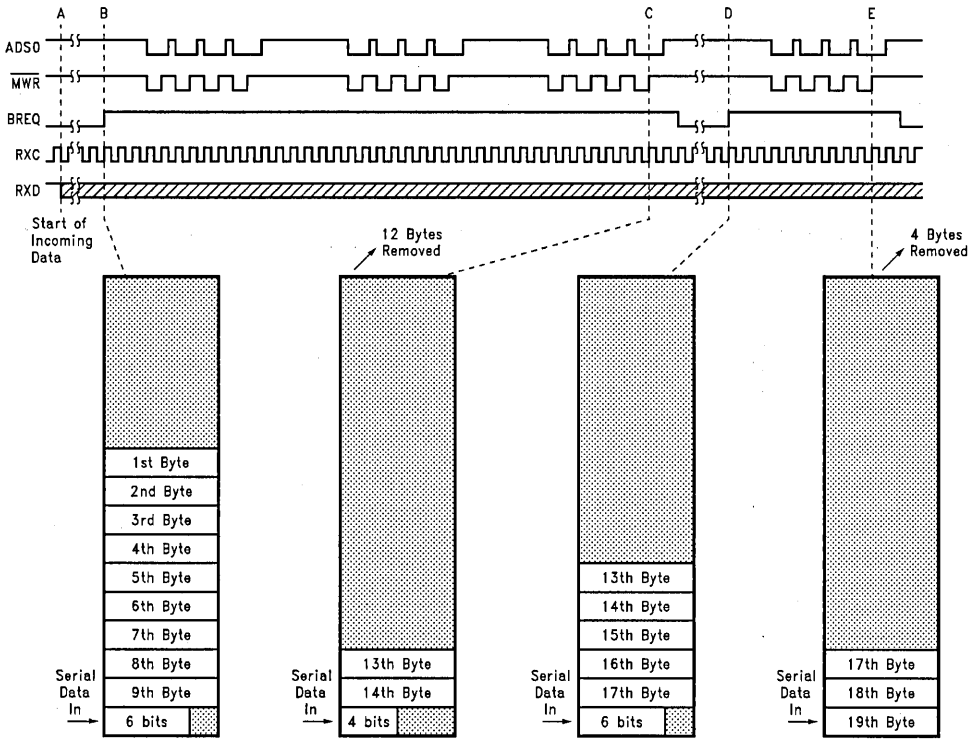


TL/F/11819-6

**FIGURE 6. Receive Packet—Byte Mode, 4 Byte Threshold—First Situation**

After the preamble and the SFD (Point A), the FIFO shifts in  $74 \pm 1$  bit of data (depending on the location of the SFD with respect to  $t_1$  of the DMA sequence) before BREQ is asserted (Point B). The FIFO transfers 2 bursts of 4 bytes each into memory while shifting in 23 bits of data until

$\overline{\text{MWR}}$  is deasserted (Point C). BREQ is asserted when there are 5 bytes and 6 bits in the FIFO (Point D). Four bytes are removed from the FIFO and stored in memory, while 10 bits are shifted into the FIFO until  $\overline{\text{MWR}}$  is deasserted (Point E). D and E are repeated until the packet is complete.

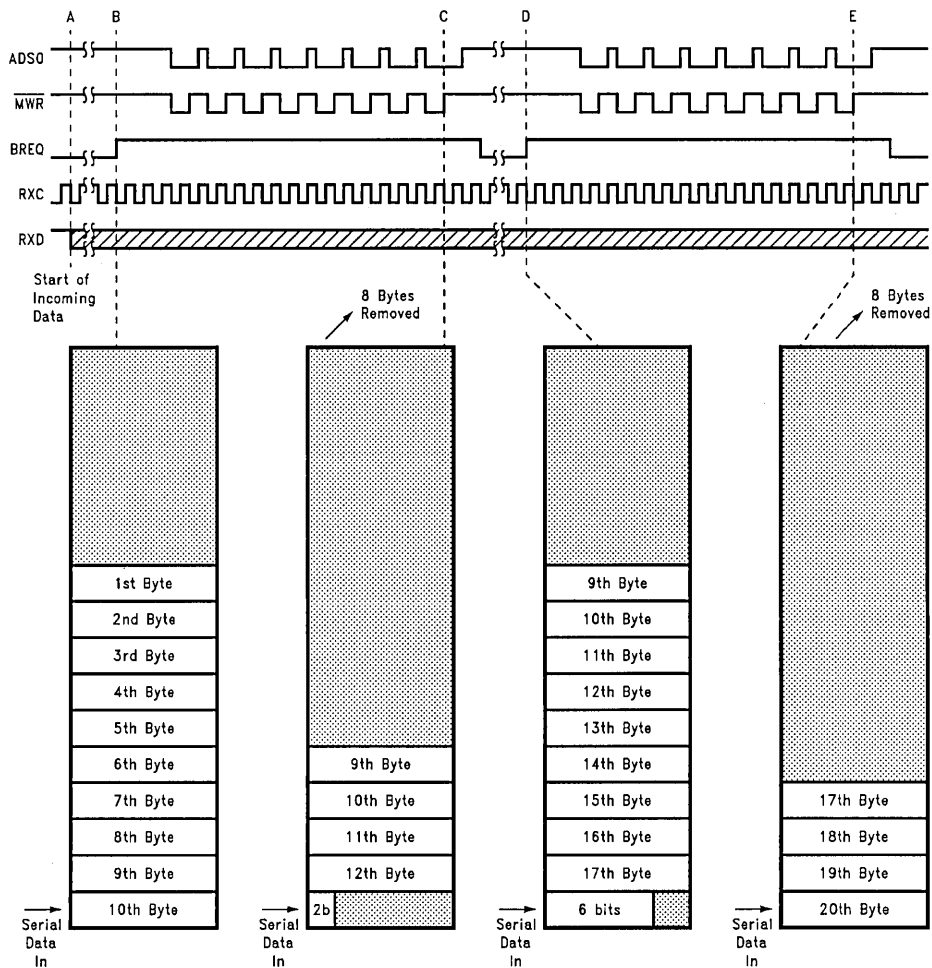


TL/F/11819-7

**FIGURE 7. Receive Packet—Byte Mode, 4 Byte Threshold—Second Situation**

After the preamble and the SFD (Point A), the FIFO shifts in  $78 \pm 1$  bit of data (depending on the location of the SFD with respect to  $t_1$  of the DMA sequence) before BREQ is asserted (Point B). The FIFO transfers 3 bursts of 4 bytes each into memory while shifting in 38 bits of data until

$\overline{MWR}$  is deasserted (Point C). BREQ is asserted when there are 5 bytes and 6 bits in the FIFO (Point D). Four bytes are removed from the FIFO and stored in memory, while 10 bits are shifted into the FIFO until  $\overline{MWR}$  is deasserted (Point E). D and E are repeated until the packet is complete.

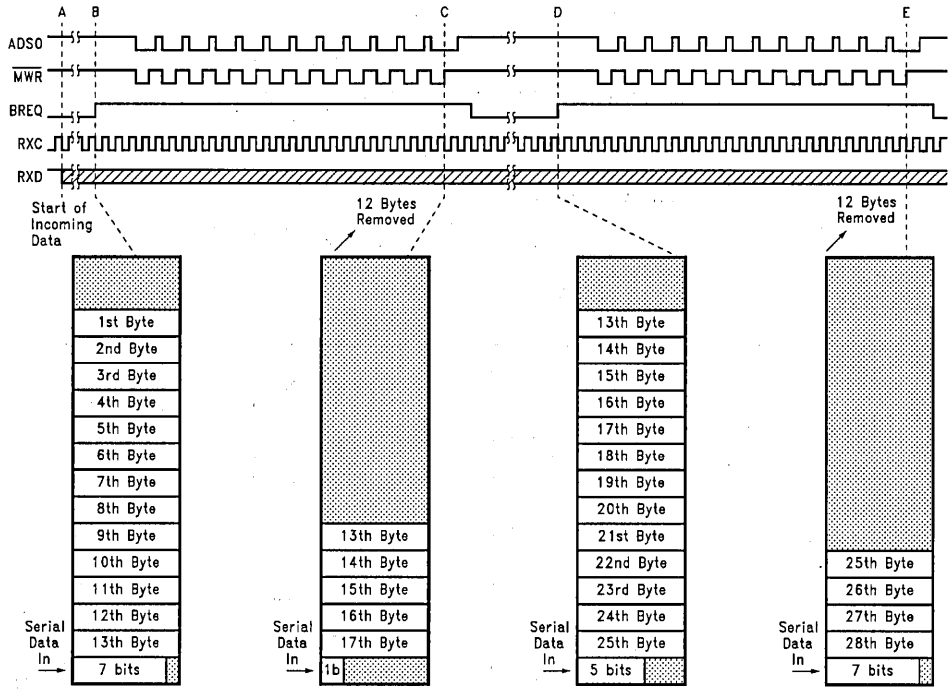


**FIGURE 8. Receive Packet—Byte Mode, 8 Byte Threshold**

TL/F/11819-8

After the preamble and the SFD (Point A), the FIFO shifts in  $80 \pm 2$  bits of data (depending on the location of the SFD with respect to  $t_1$  of the DMA sequence) before BREQ is asserted (Point B). The FIFO transfers 1 burst of 8 bytes into memory while shifting in 18 bits of data until MWR is

deasserted (Point C). BREQ is asserted when there are 9 bytes and 6 bits in the FIFO (Point D). Again, 8 bytes are removed from the FIFO and stored in memory, while 10 bits are shifted into the FIFO until MWR is deasserted (Point E). D and E are repeated until the packet is complete.



TL/F/11819-9

**FIGURE 9. Receive Packet—Byte Mode, 12 Byte Threshold**

After the preamble and the SFD (Point A), the FIFO shifts in  $111 \pm 2$  bits of data (depending on the location of the SFD with respect to t1 of the DMA sequence) before BREQ is asserted (Point B). The FIFO transfers 1 burst of 12 bytes into memory while shifting in 26 bits of data until MWR is

deasserted (Point C). BREQ is asserted when there are 13 bytes and 5 bits in the FIFO (Point D). Again, 12 bytes are removed from the FIFO and stored in memory, while 26 bits are shifted into the FIFO until MWR is deasserted (Point E). D and E are repeated until the packet is complete.



TRANSMIT PACKET

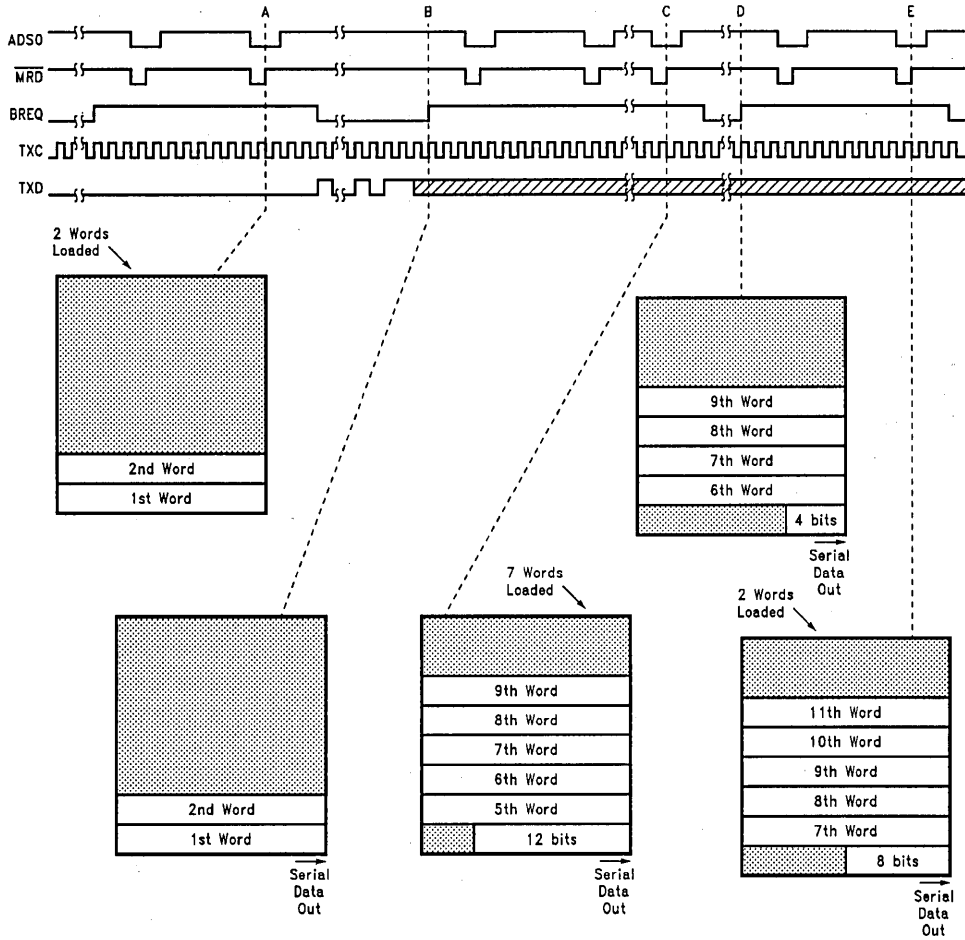


FIGURE 10. Transmit Packet—Word Mode, 1 Word Threshold

TL/F/11819-10

The FIFO prefetches 2 words from memory (point A). The preamble starts during these prefetches. After the preamble and the Start of Frame Delimiter are transmitted, data is shifted out of the FIFO, and BREQ is asserted (Point B). During this burst, 7 words are loaded into the FIFO from

memory while 52 bits are shifted out until  $\overline{MRD}$  is deasserted (Point C). The next BREQ comes when there are 4 words and 4 bits in the FIFO (Point D). Two words are loaded into the FIFO from memory while 12 bits are shifted out (Point E). D and E are repeated until the packet is complete.

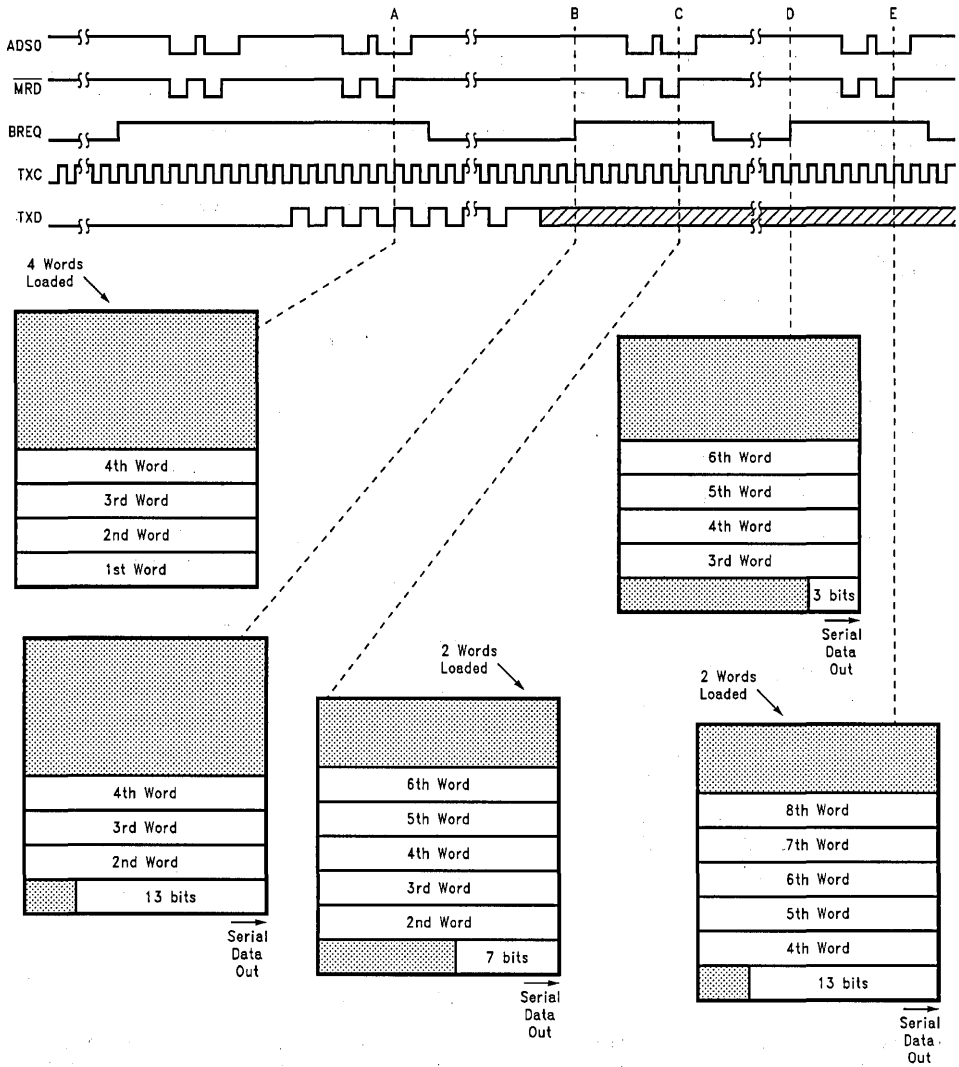


FIGURE 11. Transmit Packet—Word Mode, 2 Word Threshold

TL/F/11819-11

The FIFO prefetches 2 bursts of 2 words each from memory (Point A). The preamble starts during these prefetches. After the preamble and the Start of Frame Delimiter are transmitted, data is shifted out of the FIFO. After 3 bits are shifted out, BREQ is asserted (Point B). During this burst, 2 words are loaded into the FIFO from memory while 6 bits

are shifted out until MRD is deasserted (Point C). The next BREQ comes when there are 4 words and 3 bits in the FIFO (Point D). Again, 2 words are loaded into the FIFO from memory while 6 bits are shifted out (Point E). D and E are repeated until the packet is complete.

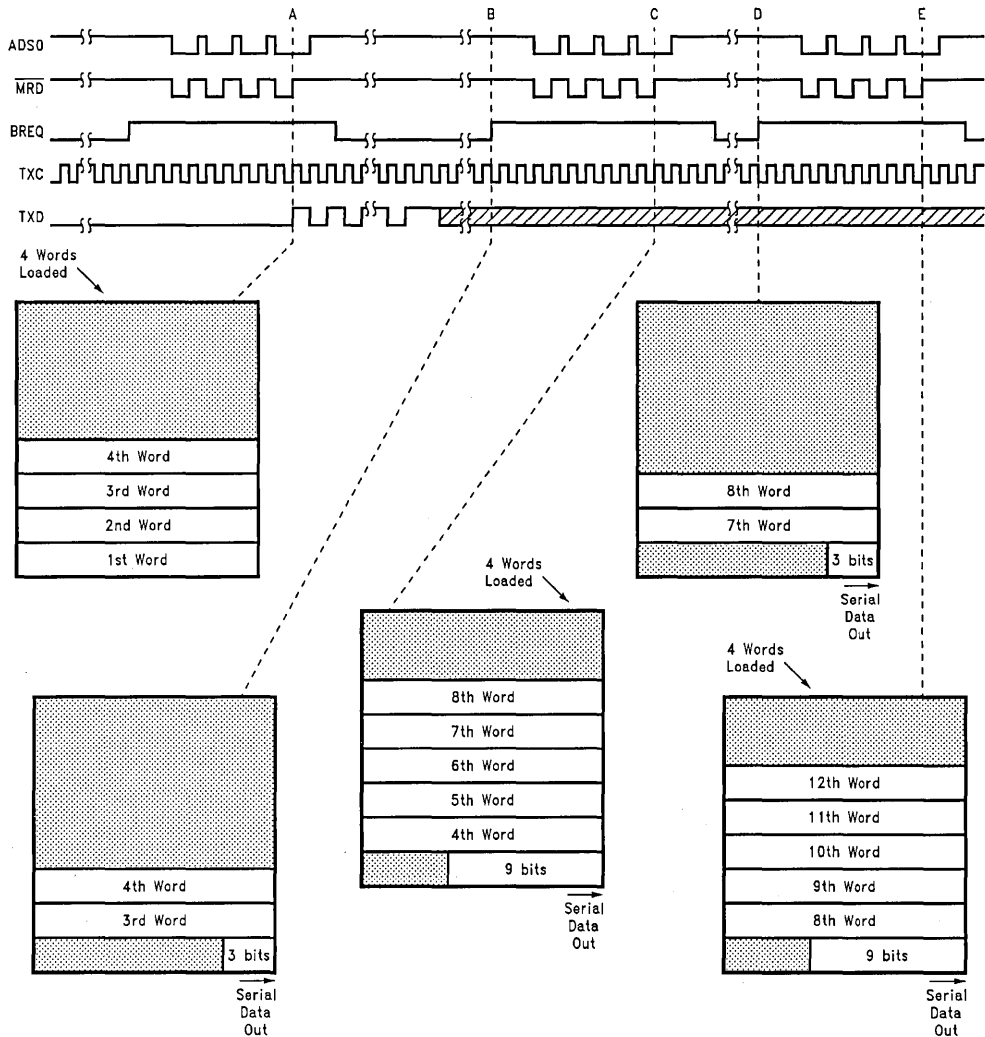
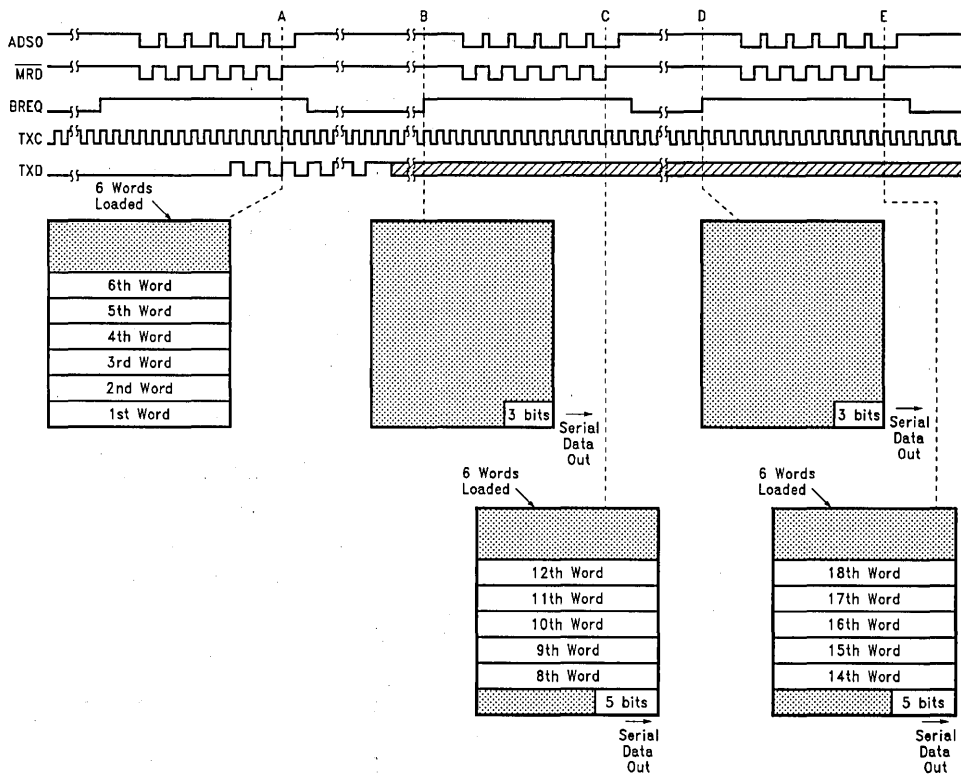


FIGURE 12. Transmit Packet—Word Mode, 4 Word Threshold

TL/F/11819-12

The FIFO prefetches 4 words from memory (Point A). The preamble starts during these prefetches. After the preamble and the Start of Frame Delimiter are transmitted, data is shifted out of the FIFO. After 29 bits are shifted out, BREQ is asserted (Point B). During this burst, 4 words are loaded into the FIFO from memory while 9 bits are shifted out until

MRD is deasserted (Point C). The next bus request comes when there are 2 words and 3 bits in the FIFO (Point D). Again, 4 words are loaded into the FIFO from memory while 9 bits are shifted out (Point E). D and E are repeated until the packet is complete.



TL/F/11819-13

**FIGURE 13. Transmit Packet—Word Mode, 6 Word Threshold**

The FIFO prefetches 6 words from memory (Point A). The preamble starts during these prefetches. After the preamble and the Start of Frame Delimiter are transmitted, data is shifted out of the FIFO. After 93 bits are shifted out, BREQ is asserted (Point B). During this burst, 6 words are loaded into the FIFO from memory while 14 bits are shifted out until MRD is deasserted (Point C). The next bus request comes when there are 3 bits in the FIFO (Point D). Again, 6 words

are loaded into the FIFO from memory while 14 bits are shifted out (Point E). D and E are repeated until the packet is complete.

**Note:** At Point B, the 7th word is latched in approximately 10 ns before its first bit is clocked out. Occasionally, 94 bits will be shifted out before the first BREQ. In this case, the first bit of the 7th word and every 8th word from then on will be corrupted. Since no error occurs, this mode should not be used.

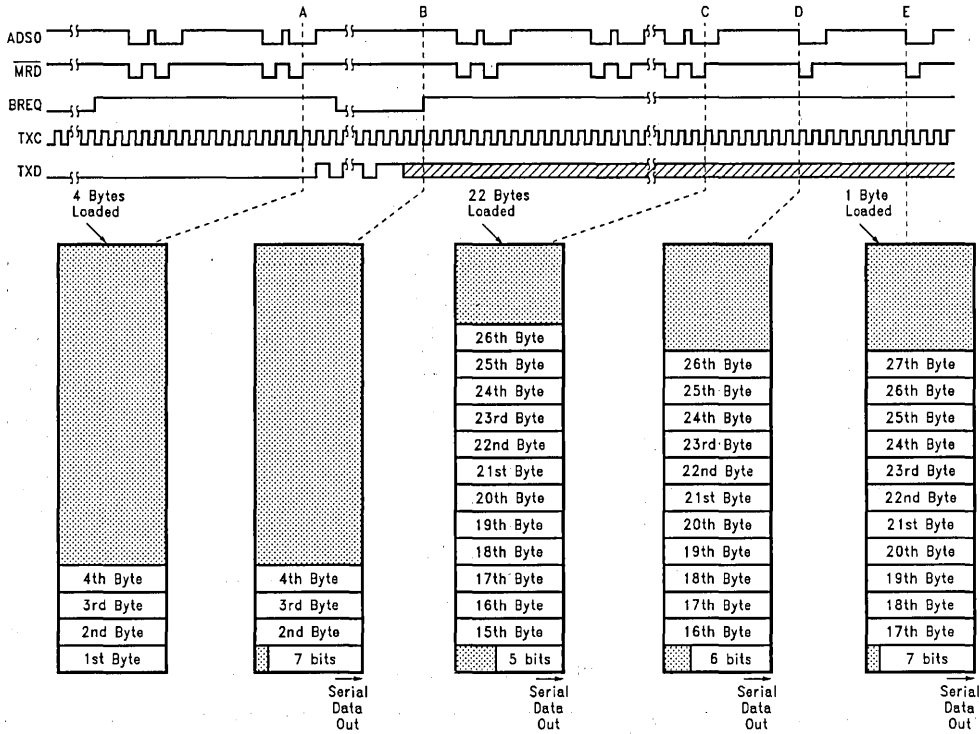


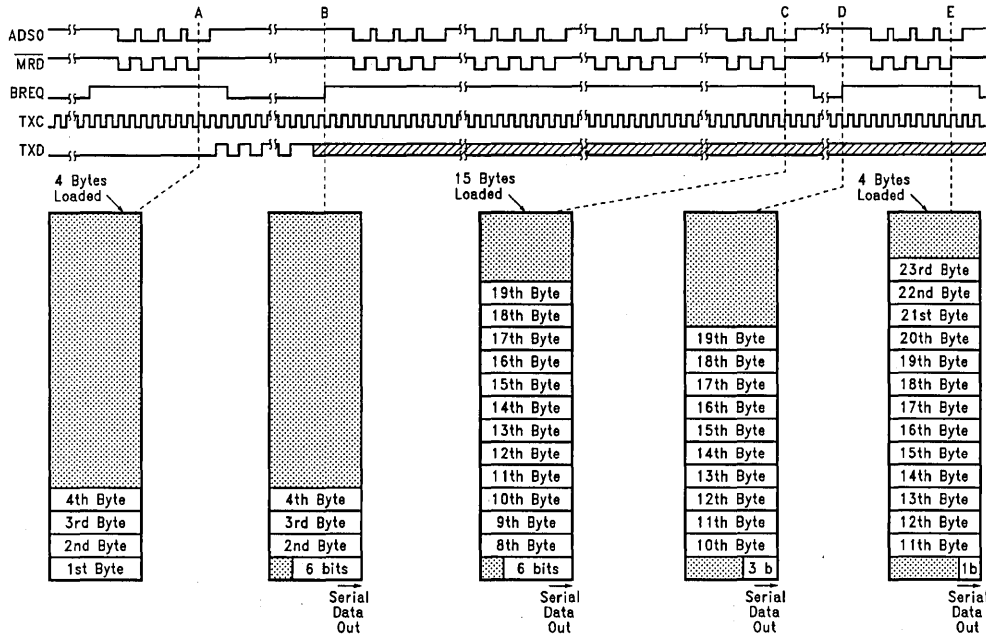
FIGURE 14. Transmit Packet—Byte Mode, 2 Byte Threshold

TL/F/11819-14

The FIFO prefetches 2 bursts of 2 bytes each from memory (Point A). The preamble starts during these prefetches. After the preamble and the Start of Frame Delimiter are transmitted, data is shifted out of the FIFO. After 1 bit is shifted out, BREQ is asserted (Point B). Eleven bursts of two bytes each are loaded into the FIFO from memory while 106 bits

are shifted out (Point C). At this point the bursts become one byte instead of two bytes (Point D). The FIFO loads 1 byte from memory while 7 bits are shifted out (Point E). D and E are repeated until the packet is complete.

**Note:** BREQ remains asserted until the entire packet is transmitted.



TL/F/11819-15

FIGURE 15. Transmit Packet—Byte Mode, 4 Byte Threshold

The FIFO prefetches 4 bytes from memory (Point A). The preamble starts during these prefetches. After the preamble and the Start of Frame Delimiter are transmitted, data is shifted out of the FIFO. After 1 bit is shifted out, BREQ is asserted (Point B). Four bursts of 4, 4, 4, and 3 bytes, respectively, are loaded into the FIFO from memory while 49 bytes are shifted out until MRD is deasserted (Point C).

The next bus request comes when there are 10 bytes and 3 bits in the FIFO (Point D). Four bytes are loaded into the FIFO from memory while 10 bytes are shifted out (Point E). D and E are repeated until the packet is complete.

**Note:** Some samples prefetches two bursts of 4 bytes each, and only 2 bursts of 4 bytes each were loaded into the FIFO between Points B and C. Both the cases sometimes had bursts of 3 bytes after point D, continuing until the packet is completed. A pattern could not be developed. This does not corrupt any bits and no error occurs.

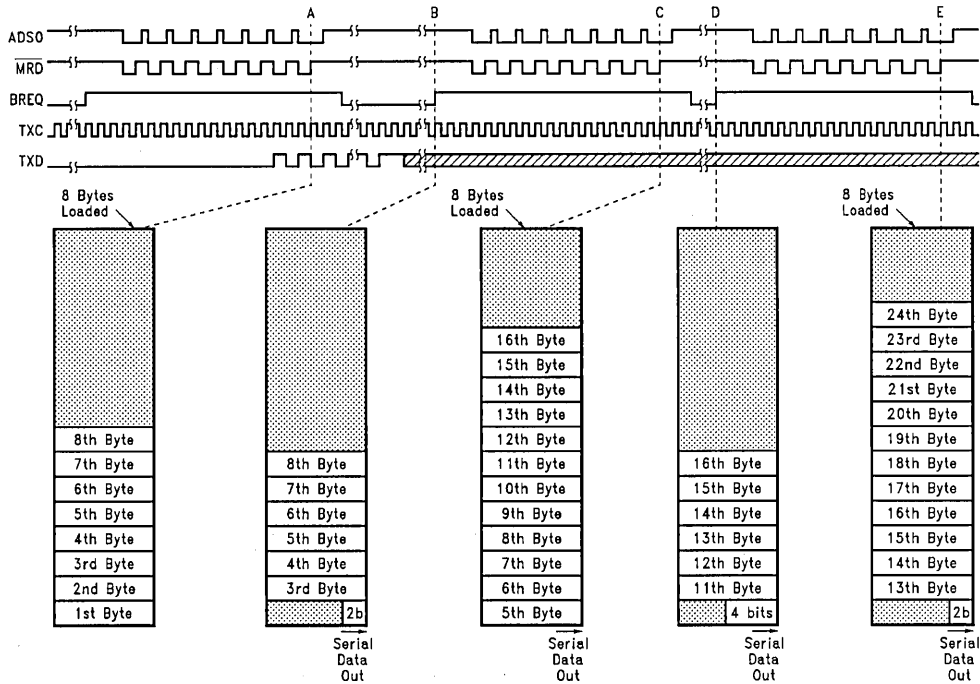
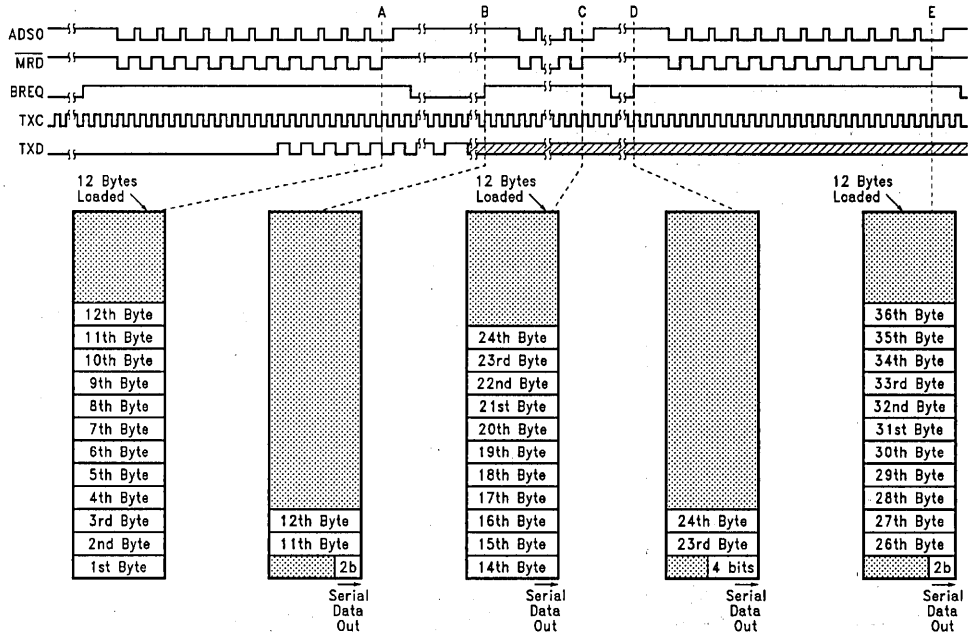


FIGURE 16. Transmit Packet—Byte Mode, 8 Byte Threshold

TL/F/11819-16

The FIFO prefetches 8 bytes from memory (Point A). The preamble starts during these prefetches. After the preamble and the Start of Frame Delimiter are transmitted, data is shifted out of the FIFO. After 22 bits are shifted out, BREQ is asserted (Point B). During this burst, 8 bytes are loaded into the FIFO from memory while 18 bits are shifted out until

MRD is deasserted (Point C). The next bus request comes when there are 6 bytes and 4 bits in the FIFO (Point D). Again, 8 bytes are loaded into the FIFO from memory while 18 bits are shifted out (Point E). D and E are repeated until the packet is complete.



TL/F/11819-17

FIGURE 17. Transmit Packet—Byte Mode, 12 Byte Threshold

The FIFO prefetches 12 bytes from memory (Point A). The preamble starts during these prefetches. After the preamble and the Start of Frame Delimiter are transmitted, data is shifted out of the FIFO. After 78 bits are shifted out, BREQ is asserted (Point B). During this burst, 12 bytes are loaded into the FIFO from memory while 26 bits are shifted out until

MRD is deasserted (Point C). The next bus request comes when there are 2 bytes and 4 bits in the FIFO (Point D). Again, 12 bytes are loaded into the FIFO from memory while 26 bits are shifted out (Point E). D and E are repeated until the packet is complete.



# AT/LANTIC™ Software Developer's Guide

National Semiconductor  
Application Note 887  
David Milne



AN-887

## INTRODUCTION

This document is designed to aid the development of software for the AT/LANTIC device. It is recommended that the AT/LANTIC data sheet be read before and then in conjunction with this description.

A brief introduction to the two Adapter architectures is given below.

## Table of Contents

### 1.0 INTRODUCTION TO THE AT/LANTIC DEVICE

#### 2.0 CONFIGURING THE AT/LANTIC

- 2.1 Changing and Saving a Configuration
- 2.2 Enabling a "New" Adapter
- 2.3 Programming Configuration Register C

#### 3.0 INITIALIZING THE AT/LANTIC

- 3.1 Hardware Reset
- 3.2 Get Bus Size/ID Bytes
- 3.3 Initializing the Registers
- 3.4 Checking the Cable
- 3.5 Checking the Interrupt
- 3.6 Checking the Boot ROM

#### 4.0 TRANSFERRING INFORMATION

- 4.1 I/O Mode Transfers
  - 4.1.1 DMA Write Sequence
  - 4.1.2 DMA Read Sequence
- 4.2 Shared Memory Transfers

#### 5.0 TRANSMISSION SEQUENCE

- 5.1 Register Sequence

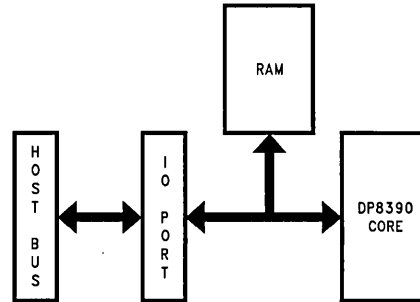
#### 6.0 RECEPTION SEQUENCE

- 6.1 The Buffer Ring
- 6.2 Removing a Packet
- 6.3 Dealing with Overflows

### 1.0 INTRODUCTION TO THE AT/LANTIC

The AT Local Area Network Twisted Pair Interface Controller provides a simple method of interfacing any ISA (Industry Standard Architecture) bus based systems to an Ethernet Network. This device can emulate one of the most popular Ethernet Adapter architectures—Novell's NE2000 adapter.

The configuration information describing the devices architecture, address, interrupt etc. can either be loaded from switches or from an EEPROM. Use of the EEPROM method allows the device to be configured solely by software thus providing a more user friendly Ethernet adapter.

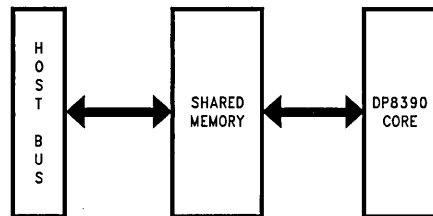


TL/F/11820-1

FIGURE 1. NE2000 Emulation Mode

The NE2000 mode utilizes a Data Port register through which all transfers to/from the buffer RAM take place. Any transfer requires the buffer RAM address, transfer size and direction to be programmed into registers before the transfer can be initiated.

This mode of operation is often referred to as I/O Mode.



TL/F/11820-2

FIGURE 2. Shared Memory Mode

In Shared Memory mode the buffer RAM is mapped into system memory. This allows any data in the buffer RAM to be directly transferred across the ISA bus.

The AT/LANTIC requires a 20H byte space in the PC's I/O Port map, this area contains all of the AT/LANTIC's registers. The contents of this register block depend on the mode the AT/LANTIC is operating in. One common factor is the NIC core register section which contains 16 registers. The location of the register block is given by "I/O address" (also referred to as I/O base). Figures 3 and 4 show the contents of the register block for either mode.

The AT/LANTIC can access a RAM area of up to 64 kbytes, however the standard is to only use 16 kbytes of this area.

1

The method of accessing this RAM area is described in Section 4.0 of this document. The AT/LANTIC's memory map depends on the mode of the device. *Figures 5 and 6* show the full 64 kbyte memory map for each mode (each mode with the chip in compatible mode utilizing only 16 kbytes for the RAM Buffer).

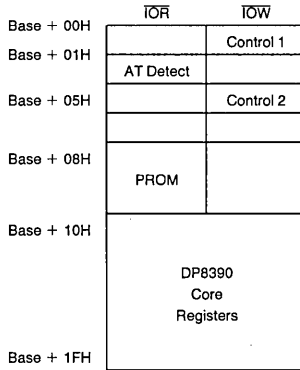


FIGURE 3. Shared Memory Mode I/O Map

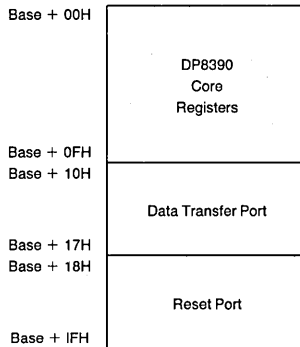


FIGURE 4. I/O Port Mode Register I/O Map

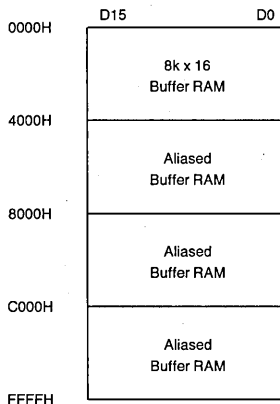


FIGURE 5. Shared Memory Mode Memory Map

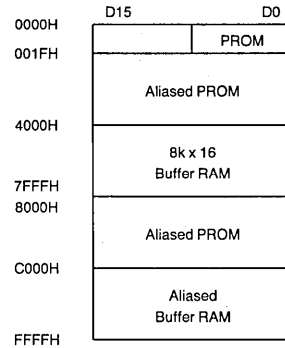


FIGURE 6. I/O Port Mode Memory Map

## 2.0 CONFIGURING THE AT/LANTIC

The AT/LANTIC controller is designed such that it is fully software configurable. The configuration information is held in three registers A, B and C as described in the data sheet under Section 5.1. As an added safety feature configuration registers A and B are hidden, so that they cannot be accidentally overwritten. Register C is only accessed during a RESET and can not be directly accessed by software.

Register A controls the I/O address, interrupt and mode of the AT/LANTIC device. Register B controls the cable type selection and certain flags altering some interface timings to the ISA bus (refer to Section 4.1 "Bus Error Condition").

### 2.1 Changing and Saving a Configuration

#### CHANGING THE CONFIGURATION

Registers A and B can be read directly at I/O address offset 0AH and 0BH respectively. To write to these registers a read access must be immediately followed by a write access. Interrupts should be disabled during the write sequence to ensure it is not corrupted. These registers can only be accessed when the NIC command register is set to page 0, refer to AT/LANTIC data sheet Section 5.3.

As register A can change the address location of the AT/LANTIC registers (and hence of reg. B) then a software update of both registers A and B should first change register B and then register A. This allows the same base I/O address to be used for both register updates.

#### THE GDLNK BIT

Special care should be taken when configuring the GDLNK bit of register B. If this bit is set to 1 then the link integrity checking (TPI mode) is disabled. If link integrity checking has not been disabled (10BT standard) then this bit reads 1 for good link and 0 for link broken. If the AT/LANTIC is in TPI mode with a good link and reg. B is read then the GDLNK bit is shown to be 1, if this was written directly back to reg. B then link integrity checking is disabled. Thus it is necessary to mask out the GDLNK bit when writing to reg. B unless the disabling of link integrity checking is required.

#### SAVING A CONFIGURATION

The AT/LANTIC has a feature allowing the required configuration to be saved to an EEPROM such that on power up the configuration registers are automatically loaded with the correct values. There is a special algorithm which writes the configuration to the EEPROM, this is described in the pseudo code below. This algorithm does not change the registers directly, i.e. the new state only appears on the next power up.

```

CONFIGURATION_SAVE ()
{
//Interrupts are disabled to ensure the
// sequence is not corrupted.
    Disable interrupts;

// Set EELOAD bit in register B.
    value = READ (Config_Reg_B);
    value = value & (~GDLNK);
    value = value | EELOAD;
    WRITE (Config_Reg_B, value);

// Output the configuration info.
    READ(Config_Reg_B);
    WRITE(Config_Reg_B, config_for_A);
    WRITE(Config_Reg_B, config_for_B);
    WRITE(Config_Reg_B, config_for_C);

// Wait for EELOAD bit to go low.
    while(value && EELOAD)
    {
        value = READ(Config_Reg_B);
        WAIT();
    }
    Enable Interrupts;
}

```

## 2.2 Enabling a "New" Adapter

It is possible to place the AT/LANTIC controller in a "disabled" state in which it shall not respond at any I/O base location. This is a particularly useful mode for software configurability as it allows the auto selection of an available configuration before the AT/LANTIC based adapter is enabled. Hence potential conflicts of Interrupt and I/O base address can be avoided.

The method of enabling the AT/LANTIC from this "disabled" state consists of writing a byte four consecutive times to port 278H, during which time interrupts should be disabled to ensure the sequence is not corrupted. The lower 3 bits of this byte inform the AT/LANTIC of which address it should enable to. More information on the mapping of these three bits to I/O base locations can be found in the AT/LANTIC data sheet under Section 5.1.

The port 278H is normally a PC's secondary printer port. Using the "four writes" sequence ensures that an adapter is not accidentally enabled if the port is in use. If the port is active when the adapter is to be enabled then it is possible to corrupt a print sequence, to avoid this the code should check if the port is active and if so wait until the port is free. The printer uses port 278H as a data port and port 279H as a control port, reference should be made to the PC's technical manual for an explanation of these registers and how the port operates.

Once the AT/LANTIC is enabled configuration register A may contain old information in bits 3-7. Register A bit 7 is the MEMIO (architecture) bit, the offset of the configuration registers varies depending on the state of this bit. Thus the software has to detect which architecture mode the AT/LANTIC has appeared in before bits 3-7 of reg. A and all of reg. B can be overwritten with the new configuration information.

## DETECTING AT/LANTIC MODE

The AT/LANTIC can appear in either the I/O port mode or the Shared Memory mode. As the I/O address is known then the mode of operation can be found by checking the data at the address of register A, i.e., check offset 0AH for I/O mode and offset 1AH for Shared Memory mode. The recommended method of detecting the architecture of an AT/LANTIC at a known I/O base is given in the pseudo code below.

```

FIND_MODE()
{
// Check if in I/O mode
    config_a = READ(IO_BASE + 0AH);
// Check MEMIO is low i.e. I/O mode.
    if((config_a & 80H) == 0)
    {
// Check IOAD bits match the I/O base addr.
        if((config_a & 7) relates to IO_BASE)
            return(IO_MODE_DETECTED);
    }
// Check if in Shared Memory mode.
    config_a = READ(IO_BASE + 1AH);
// Check MEMIO is high i.e. S/M mode.
    if((config_a & 80H) == 1)
    {
// Check IOAD bits match the I/O base addr.
        if((config_a & 7) relates to IO_BASE)
            return(SHARED_MEM_DETECTED);
    }
// No AT/LANTIC mode detected.
return(NO_MODE_DETECTED);
}

```

## 2.3 Programming Configuration Register C

Configuration register C can not be accessed directly by software. Details on what configuration register C controls can be found in the AT/LANTIC data sheet Section 5.1. The upper four 4 bits of register C are fixed depending on the design of the adapter card, the lower four bits vary depending on the boot ROM option selected. It is necessary to know the boot ROM option selected so that register C is correctly updated by the "Configuration\_Save" routine (as given under Section 2.1). As register C can not be read some mechanism is required to detect the boot ROM option in use. The recommended method is to place a signature text string in the boot ROM at a fixed location. This string would contain information on the size of ROM in question. The code should then scan the RAM space for this signature string. If found, then the location and size of the boot ROM is known and the value required for register C can be calculated. As register C can not be written to directly, the only method of updating it is to use the "Configuration—Save" routine.

It should be noted that any change to the value of register C is not active until the AT/LANTIC has been reset (normally a power off/on of a PC) and the new contents of the EEPROM have been loaded into the registers.

### 3.0 INITIALIZING THE AT/LANTIC

The following section deals with all of the functions necessary to initialize and partly test the AT/LANTIC device. It is recommended that the code follows the order of each of the sub-sections.

#### 3.1 Hardware Reset

The first step in the initialization sequence is to provide a reset pulse to the NIC core of the AT/LANTIC device. The method of providing this signal depends on the architecture selected for the AT/LANTIC.

#### I/O MODE RESET SEQUENCE

In this mode a portion of the I/O address map acts as a reset port, offsets 18H to 1FH, any of these offsets can be used as the reset port. To activate a reset the port should be read from and then written to (with any value). Following this a delay of 1.6 ms is required to make sure the reset has completed.

#### SHARED MEMORY RESET SEQUENCE

This mode contains two control registers (refer to Section 5.2 of the AT/LANTIC data sheet). The MSB of control register 1 is the RESET flag. To activate the reset this bit should be toggled high then low. Again a delay of 1.6 ms is required to allow the reset to complete.

#### PLACING THE AT/LANTIC IN STOP MODE

Following the hardware reset it is necessary to place the NIC core of the AT/LANTIC in the STOP mode, which performs a software reset. This is done by setting the STOP and RS2 bits of the NIC command register (refer to the AT/LANTIC data sheet Section 5.3 for more information on the NIC core registers), PS0 and PS1 are cleared to ensure that the NIC core is in "page 0". (The NIC core registers are split into 3 pages of 16 registers, the bits PS0 and PS1 in the command register define which page the NIC is currently operating in.) The code should wait until the software reset is indicated as complete (when the RST bit of the Interrupt Status register is set to a 1).

#### 3.2 Get Bus Size/ID Bytes

This section deals with detecting the size of the slot an adapter has been inserted into and checking ID byte values are correct for the mode of operation selected. The two AT/LANTIC modes are quite different in their approach to this problem and shall be discussed separately.

#### I/O MODE

The I/O mode architecture maps PROM data into RAM space locations 00H to 1FH. The contents of this RAM space are given in *Figure 7*. Offset 1EH contains a word value which depends on the size of slot in which the adapter currently resides. If the value equals 5757H, then the slot is 16-bit, if the value equals 4242H, then the slot is 8-bit. If the value does not equal either of the previous values, then the adapter is not correctly functional in NE2000 emulation mode, i.e., these bytes also act as ID bytes. An example of the DMA read operation required to get the word from offset 1EH is described in Section 4.1.2 of this document. The initial transfer requires the Data Configuration Register (DCR) to be programmed for 8-bit operation, and the PC to perform two byte wide reads of the Data Port. Once the data has been transferred the DCR and PC transfers can be set for the correct bus width.

#### SHARED MEMORY MODE

The shared memory mode contains a register which holds information on the size of slot occupied by the adapter. The LSB of this register, the AT detect register (refer to AT/LANTIC data sheet Section 5.2) is set to 1 for 16-bit mode and 0 for 8-bit mode.

Correct emulation of the Ethercard Plus16 requires an ID byte. In this mode the registers at offsets 08H to 0FH (see *Figure 8*) contain first the Ethernet address then an ID byte followed by a checksum byte.

For the AT/LANTIC shared memory mode the ID byte at offset 0EH should contain value 05H. The two complement sum of all of the eight bytes should equal FFH. If this is not the case, then the adapter is not correctly operating in Ethercard Plus16 emulation mode.

00H	ETHERNET ADDR. 0
01H	ETHERNET ADDR. 1
02H	ETHERNET ADDR. 2
03H	ETHERNET ADDR. 3
04H	ETHERNET ADDR. 4
05H	ETHERNET ADDR. 5
06H	(BOARD ID BYTE)
07H	(CHECKSUM)
1EH	42H or 57H
1FH	42H or 57H

FIGURE 7. NE2000 Mode PROM Memory MAP

0BH	Addr. Byte 0
	Addr. Byte 1
	Addr. Byte 2
	Addr. Byte 3
	Addr. Byte 4
	Addr. Byte 5
	Board ID Byte
0FH	Checksum

FIGURE 8. S/M Mode PROM Loaded Registers

#### 3.3 Initializing the Registers

When initializing an adapter for Shared Memory mode it is necessary to set up two control registers before initializing the NIC core registers, the following step is ignored for I/O Port mode.

#### SHARED MEMORY ADDRESS INITIALIZATION

In shared memory mode the Buffer RAM is mapped into the PC address space. Two control registers define where in the PC memory map this RAM shall appear, refer to Section 5.2 of the AT/LANTIC data sheet. Address bits A13–A18 of the selected memory address should be programmed into bits 0–5 of control register one. Address bits A19–A23 should

be programmed into bits 0–4 of control register two. The MEMW bit of control register 2 defines if the memory is 8 kwords or 8 kbytes in size. This bit should be defined during this initialization. It may also be necessary to initialize the MEME bit of control register 1, this depends on the method of controlling the buffer RAM—refer to Section 4.2.

- e.g. D0000H, 16 kbytes memory.
- Control reg. 1 is 28H,
- Control reg. 2 is 41H.

**NIC CORE REGISTER INITIALIZATION**

The following register initialization sequence is mandatory for both I/O Port and Shared Memory modes. All of the registers discussed are explained in detail under Section 5.3 of the AT/LANTIC data sheet.

1. Put the AT/LANTIC in STOP mode. Refer to Section 3.1 on putting the AT/LANTIC in STOP mode.
2. Initialize Data Configuration Register (DCR), WTS (transfer width) dependent on the result of the previous section, LS set, ARM is dependent on the method chosen to control the receive buffer ring (refer to Section 6.0). The FIFO threshold is typically set to 8 bytes/4 words i.e., FT1 is set. (When in Shared Memory mode the WTS bit can always be set if 16 kbytes of RAM are being used.)
3. Clear Remote Byte count registers.
4. Initialize Receive Configuration Register (RCR), this register determines which packets are accepted by the AT/LANTIC and buffered into RAM. The options available are save errored packets, runt packets, multicast packets, broadcast packets, physical address match packets and all physical address packets (promiscuous mode). Setting the register to 00H allows only physical address match packets. Further discussion on setting the physical and multicast addresses is given later in this section.
5. Place the NIC in loopback mode 1 or 2 by setting bits LB0 or LB1 in the Transmit Configuration Register. Loopback is described in the data sheet under Section 6.5.
6. Initialize the Receive Buffer Ring, Boundary Pointer (BNDRY), Page Start (PSTART), Page Stop (PSTOP), Transmit Page Start (TPSR).

The values for these registers are discussed at the end of this section.

7. Clear the Interrupt Status Register (ISR), by writing FFH to it.
8. Initialize Interrupt Mask Register (IMR), this register controls which sources of interrupt are allowed or disallowed. It would be normal to set PRXE (packet received), PTXE (packet transmitted), PTXEE (packet transmission error) and OVWE (overflow) in the register.
9. Program the Command Register for page 1 (set bits PS0, RD2 and STP) and initialize the physical, multicast and CURR registers as described later in this section.
10. Put the NIC in to START mode, set the START bit and clear the STOP, PS0 and PS1 bits in the Command Register. The receive is still not active as the NIC core is in loopback mode.

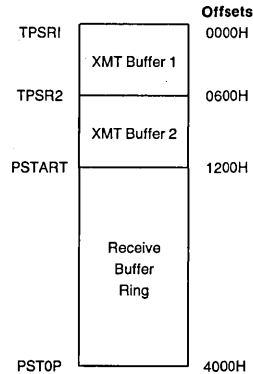
11. Remove the AT/LANTIC from loopback mode by initializing the Transmit Configuration Register (TCR) to its correct value, typically 00H.

The AT/LANTIC is now ready to receive and transmit packets.

**SETTING UP THE BUFFER RING**

The values to be programmed in to the buffer ring pointers PSTART, PSTOP, CURR, BNDRY and TPSR depend on the mode of operation of the AT/LANTIC. There are several possible modes NE2000 and Ethercard Plus16 emulation modes which use 16k of Buffer RAM and a non-compatible mode which allows up to 64k of Buffer RAM. The size of transfers in question also alter the size of the Buffer Ring.

The buffer RAM is split into "pages" each containing 256 bytes. The standard is to have a transmit buffer followed by the receive buffer ring. It is recommended that two transmit buffers be utilized as this improves performance. One packet can be loaded into the RAM while another is being transmitted on the network. Each transmit buffer requires 6 "pages" for a full Ethernet packet (some protocols may not require a maximum Ethernet packet). The Transmit Page Start Register (TPSR) should point to the buffer being transmitted for the whole duration of a transmission, i.e., when the alternate buffer is being loaded during a transmission the TPSR should not be altered. The receive buffer ring follows the transmit buffer. The Page Start register (PSTART) is set to the page following the transmit buffer(s). The Page Stop (PSTOP) register is set to the end of the buffer RAM plus one page (the size of buffer RAM is determined by the bus width/memory width.)



**FIGURE 9. Buffer RAM Layout**

In NE2000 emulation mode the buffer RAM resides at locations 4000H to 8000H (refer to Figure 6). The Ethercard Plus16 buffer RAM resides at 0000H to 4000H (refer to Figure 5). These examples are for 16-bit modes.

There are two possible methods of controlling the receive buffer ring a software method and an automated method called "send packet". These are discussed in more detail in Section 6.0 of this document.

Initializing CURR and BNDRY for

- (i) the software method,
- CURR = PSTART + 1
- BNDRY = PSTART
- Next\_PKT = PSTART + 1

(ii) the "send packet" method,  
 CURR = BNDRY = Next\_PKT = PSTART  
 ("Next\_PKT" is a software declared variable further described in Section 6.0).

#### SETTING THE PHYSICAL ADDRESS REGISTERS

During initialization, the NIC core physical address registers are loaded with the adapters Ethernet address (refer to Section 5.3 of the AT/LANTIC data sheet). Each adapter contains a unique six byte address for identification on a network. The Ethernet address is held in a PROM store in the AT/LANTIC, the method of retrieving the information depends on its present architecture mode. In shared memory mode the Ethernet address is held at register offsets 08H to 0DH, see *Figure 3*. These values can then be written to the NIC physical address registers. In I/O mode the address is held in the first three words of the RAM thus a Remote DMA read is required to retrieve the information, refer to Section 4.1.2.

#### SETTING THE MULTICAST REGISTERS

To allow a network station to receive packets destination addresses other than the stations physical node address, it is necessary to store a list of these destination addresses. A group of addresses to be received are referred to as multicast addresses. This device can not hold all of the addresses, thus the AT/LANTIC contains 8 multicast address registers (MAR0-7) which decode the addresses to be received. These multicast registers provide filtering of multicast addresses hashed by the CRC logic. All destination addresses are fed through the CRC logic and as the last bit of the destination address enters the CRC, the 6 MSB's of the CRC generator are latched. These six bits are then used to index a unique filter bit (FB0-63) in the multicast address registers. When a software developer wishes to accept a specific multicast address the above sequence should be followed to determine which filter bit in the multicast registers should be set. Several bits can be set to accept several multicast addresses. A pseudo code example of the routine required for this is given below.

```
// Hexadecimal equivalent of the NIC's CRC
// equn.
define CRC_POLYNOMIAL 04C11DB6H

// The multicast address is held in a 6 byte
// array.
unsigned char mult_addr[6];
crc = FFFFFFFFH;

// The following loops create the 32-bit CRC
// value.

// Loop through each byte of the address.
for(i=0; i<6; i++)
{
// Loop through each bit of that byte.
for(bit=0; bit<8; bit++)
```

```
{
  carry=(crc31)^(mult_addr[i] &
    (1<<bit)) > bit);
  crc <= 1; if (carry)
    crc = ((crc^CRC_POLYNOMIAL) | carry);
}
}

// Extract the 6 MSB's from the CRC value,
// this six bit value is used to index a
// unique filter bit.
index >= 26;
index &= 3FH;
// Find the multicast register number and
// bit of that register to set.
register_no = crc > 3;
register_bit = 1 < (crc & 7);

// Calculate the new register value.
value = READ(MAR[register_no]) |
register_bit;

// Set the Multicast Address Register (MAR)
// value.
WRITE (MAR[register_no], value);

3.4 Checking the Cable

There are four possible media types that can be selected
with the AT/LANTIC device. Both the TPI(10BT) and
TPI(non spec.) modes simply use the GDLNK bit of configura-
tion register B to indicate the cable status. The test for a
good "Thin" Ethernet cable is more involved and a pseudo
code description of what is required is given below. Thick
Ethernet cable can be checked by performing level 3 loop-
back as described in Section 6.5 of the AT/LANTIC data
sheet. However this test can only be guaranteed if per-
formed on a non-active network, i.e. no information is being
passed on the network during the test.

// Assume already initialized.

thin_cable_check()
{
// Set up a cable check packet for
// transmission. The destination addr.
// should equal the source address to avoid
// other stations receiving this pkt. The
// data field is of the developers choice.
  packet = set_up_cable_pkt();
  length = size of (packet);
```

```

// Set the transmit byte count registers.
WRITE(TBCRO, length_LSB);
WRITE(TBCR1, length_MSB);

// Clear the Interrupt Status Register.
WRITE(INTSTATUS, FFH);

// Issue the transmit command.
WRITE(CMND, (RD2 | STA | TXP));

// Poll the interrupt status register until
// the packet is indicated as transmitted
// or there is a timeout.
while(time_in_loop < 1 second)
{
    STATUS = READ(INTSTATUS);

    if(STATUS & (ISR_TXE | ISR_PTX))
        break;

    update--time--in--loop();
    Short_Delay();
}

// If the routine timed out the tx failed.
if(time_in_loop > 1 second)
    return(NO_CABLE);

// Read the Transmit Status Register.
TSR_value = READ(TSR);

// Check if there were excessive collisions.
if(TSR_value & ABT)
    return(UNTERMINATED);

// If there was 1 to 15 collisions the cable
// is good.
if(TSR_value & COL)
    return(CABLE_OK);

// Check for other transmission failures
// only after collision check because if a
// collision occurred it can set some of
// the following bits in error.
if(TSR_value & (CDH | CRS | FU));
    return(NO_CABLE);

// If this point is reached the tx passed.
return(CABLE_OK);
}

```

More detail on the Interrupt Status Register (ISR) and the Transmit Status Register (TSR) bits can be found in the AT/LANTIC data sheet under Section 5.3.

### 3.5 Checking the Interrupt

When an interrupt is detected an interrupt handler is called. This handler should then investigate the cause of the interrupt and act accordingly. In the case of the AT/LANTIC there is an Interrupt Status Register which provides information on the cause of the interrupt. It is necessary for the handler to be installed before this test is carried out.

The following pseudo code routine checks interrupt operation by assuming that when the interrupt handler is called and the ISR is found to have its PTX bit set then a "packet\_transmitted" flag is incremented/set.

```

// Assume Initialized and ISR installed.
Interrupt_check()
{
    // Clear the 'packet_transmitted' flag.
    packet_transmitted = 0;

    // Clear the Interrupt Status Register.
    WRITE(INTSTATUS, FFH);

    // Set the transmit byte count zero.
    WRITE(TBCRO, 0);
    WRITE(TBCR1, 0);

    // Issue the transmit command.
    WRITE(CMND, (RD2 | STA | TXP));

    // This transmission is used to generate an
    // interrupt.

    // Loop until ISR is called and
    // packet_transmitted flag is set
    // or there is a time out.
    while(time_in_loop < 1 second)
    {
        if(packet_transmitted)
            return(Interrupt_OK);

        update_time_in_loop();
    }
    return(Interrupt_FAILED);
}

```

### 3.6 Checking the Boot ROM

If a boot ROM has been identified as belonging to the adapter under going diagnostics then it is possible to check that its data has not been corrupted. The sum of all of the ROM bytes should equal 0. The size of the ROM (in 1/2 k segments) is held at byte offset 03H, offsets 00H and 01H should hold values 55H and AAH respectively.

#### 4.0 TRANSFERRING INFORMATION

Before transmission of a packet or processing of a reception, it is necessary to store or retrieve data from the buffer RAM. The two architecture modes available on the AT/LANTIC have different methods of accessing the buffer RAM.

##### 4.1 I/O Mode Transfers

When in I/O mode the NIC core of the AT/LANTIC transfers data to or from the buffer RAM using one of its DMA channels. The software programs the start address of the RAM segment to be transferred, the size of transfer and the direction of the transfer. The DMA controller passes data between the data transfer port and the buffer RAM, byte or word at a time. The system always writes or reads data via this port. The Data Transfer Port is mapped from I/O base offset 10H to 17H (any of these registers act as the Data Transfer Port). It is important that no other value be written to the command register during a DMA.

##### BUS ERROR CONDITION

On some implementations of the ISA bus it is possible for an error condition to arise during a DMA transfer. This is flagged by a "BE" (Bus Error) bit in configuration register B being set (writing a one to this bit resets it). The AT/LANTIC provides two differing methods for correcting this condition, these are implemented by setting either the "IO16CON" bit or the "CHRDY" bit in configuration register B. For a more detailed discussion on the cause of the error and the remedies available reference should be made to the AT/LANTIC data sheet Section 6.7 ("16-Bit I/O Cycles with CHRDY Fix").

It is recommended that a check for this error condition be carried out before an adapter is enabled on a network. This can be done by performing a DMA, checking if the "BE" bit has been set and if so implementing one of the fixes. It is also possible to perform the check at the end of every DMA (as the error may not happen during all DMA sequences).

##### 4.1.1 DMA WRITE SEQUENCE

The DMA write sequence is typically used to load up a packet to be transmitted in to the transmit buffer, as in the example below.

```

...
// Create a transmit packet and hold a
// pointer to its address in the PC RAM.
packet = set_up_xmt_pkt();

Disable Interrupts;

// Write out the Buffer RAM address for the
// xmt packet to the Remote DMA addr.
// registers. This is either TPSR1 or TPSR2.
WRITE(RSAR0, TPSR_LSB);
WRITE(RSAR1, TPSR_MSB);

// Write out the size of the packet to the
// Remote DMA BYTE count registers.
length = SIZEOF(packet);
WRITE(RBCR0, length_LSB);
WRITE(RBCR1, length_MSB);

```

```

// Issue the Remote DMA write command.
WRITE(CMND, (RDL | STA));

// Loop until all bytes/words transferred.
address = packet;
for(loop through the transmit pkt)
{
value = contents of address;
WRITE(DATA_PORT, value);
increment address pointer;
}

// It is necessary to wait until the last
// transfer is flagged as being placed into
// memory. An access to the command
// register before the DMA has completed may
// corrupt the last transfer and lead to
// serious system errors.
while(time_in_loop < 1 second)
{
status = READ(INTSTATUS);

if(status & ISR_RDC)
return(TRANSFER_OK);

update_time_in_loop();

Short_Delay();
}

Enable Interrupts;

return(TRANSFER_FAILED);
...

```

##### 4.1.2 DMA READ SEQUENCE

The DMA read sequence is typically used for removing packets from the receive buffer ring. However the example below reads the Ethernet address from Buffer RAM (as would be required during initialization).

```

...
// Read the Ethernet address and place it at
// PC RAM space pointed to by 'addr'.

Disable Interrupts;

// Set the Remote read start address to 0.
WRITE(RSAR0, 0);
WRITE(RSAR1, 0);

```



```

// Set the Remote DMA BYTE count to 6.
WRITE(RBCRO, 6);
WRITE(RBCRL, 0);

// Issue the Remote DMA read command.
WRITE(CMND, (RDO | STA));

// Read 3 words or 6 bytes depending on bus
// width.
for(loop 3 or 6 times)
{
  contents of addr = READ(DATA_PORT);
  increment address;
}

Enable Interrupts;

return(TRANSFER_COMPLETED);
...

```

The above examples often refer to the Command and Interrupt Status registers and their associated bits. More details on these registers can be found in the AT/LANTIC data sheet Section 5.3.

The size of access to the data port is dependent on the width of the bus detected during the initialization sequence. However, it should be noted that in 16-bit mode the length of transfer is programmed in bytes even if the transfers to/from the data port are to be word wide.

There are some assembly language commands that greatly simplify the transfer loop to the data port.

e.g., for the Intel 286 Processor.

```

set cx to byte count;
set es:di to pc RAM destination;
set dx to DATA_PORT;
set direction flag;
rep insw/outsw

```

#### 4.2 Shared Memory Transfers

When in Shared Memory mode the buffer RAM is mapped into a portion of the PC RAM space. Access to the buffer uses the same method as access to any portion of the PC address space. The location at which the buffer RAM appears is controlled by two Shared Memory Control Registers. These registers also set the size and width of transfer as well as enabling or disabling the RAM buffer. The buffer RAM can be set to 16 kbytes or 8 kbytes in size (as determined by the adapter hardware) and the CPU transfer width can be 8-bit or 16-bit wide (as determined in Section 3.2). These settings are controlled by two register bits, the MEMW bit of control register 2 controls the width of memory i.e., 8 kbytes/16 kbytes and the 8-bit/16-bit of control reg. 2 controls whether the transfer is 8-bit or 16-bit wide. If the transfers are to be 16-bit wide then the 8-bit/16-bit should be set only for the duration of the transfer. It is also possible to disable the buffer RAM when there is no transfer in progress by using the MEME bit of control register 1. This allows more than one adapter to utilize the same RAM location in the PC memory map. It should be noted that if multiple adapters are to use the same RAM location then all the adapters must disable the RAM when transfers are not in

progress. If only one adapter is allowed to use the RAM location then the memory can be enabled at initialization.

If the software being developed has to be fully compatible with the Ethercard Plus16 architecture then the control registers cannot be assumed to be readable. Thus when toggling the MEME and 8-bit/16-bit during a transfer it is necessary to either re-calculate the value of the address bits in the register or recall some store of the initialized value.

#### EXAMPLE TRANSFER

This example is for 16 bit wide transfers with memory only enabled for the duration of the transfer.

```

// 'Cntrl1' and 'Cntrl2' are stored at
// initialization.

// Enable the buffer memory.
WRITE(CNTRL1, (cntrl1 | MEME));
// Enable 16-bit wide transfers.
WRITE(CNTRL2, (cntrl2 | 8/16));

// Memory transfer.
MEMCOPY(PCaddr, BufferAddr, size);

// Disable the buffer memory.
WRITE(CNTRL1, cntrl1);
// Disable 16-bit wide transfers.
WRITE(CNTRL2, cntrl2);

```

There are some assembly language commands that provide a simple and efficient means of doing the "memcpy()" function in the above example.

e.g., for the Intel 286 Processor.

```

set cx to byte count;
set es:di to pc RAM destination;
set ds:si to pc RAM source;
set direction flag;
rep movsw

```

#### 4.3 The Boot ROM

The AT/LANTIC provides support for both standard boot ROMs and FLASH PROMS. Configuration register C controls the location at which the ROM is enabled, access to the ROM is then the same as access to any area of PC RAM.

The use of a FLASH prom allows software to directly update a boot ROM with the latest driver software for an adapter, thus eliminating the need to replace a ROM for each new release of software. It is necessary to set the BPWR bit of configuration register B when a FLASH prom is to be updated. Write cycles to the ROM area are only allowed when this bit is set. After this bit is set the software should follow the algorithm to program the FLASH prom (as given in the PROM's data sheet).

#### 5.0 TRANSMISSION SEQUENCE

When a transmission is required, it is necessary to select a free transmit buffer to place the packet information into (assuming there is more than one transmit buffer in the buffer RAM). The packet should then be transferred, as described in Section 4.0, to this location. A check is then required to see if a packet is presently being sent out onto the network. If a transmission is in progress then the address and size of

this new packet should be held in a "store" until the transmission completes. If a transmission is in progress then this packet can be sent directly out on to the network. This requires the Transmit Page Start Register and the Transmit Byte Count Registers to be programmed with the address and length of the packet respectively. Following this the transmit command can be issued to the Command Register. It is important to remember that the TPSR cannot be altered during a transmission.

#### ISSUING A TRANSMIT

```

...
// No transmission in progress. The new
// packet is held at transmit buffer
// address XMT_BUFFERx
    WRITE(TPSR XMT_BUFFERx_MSB);
// Set the byte size of the packet, held in
// 'length'.
    WRITE(RBCR0, length_LSB);
    WRITE(RBCR1, length_MSB);

// Issue the transmit command.
    WRITE(CMND, (RD2 | STA | TXP));

```

#### 5.1 Transmission Status Checking

Once a transmission has completed the AT/LANTIC will return an interrupt. It is the task of the Interrupt Service Routine (ISR) to determine whether the transmission was successful or not and deal with it accordingly. If the transmission completed successfully, a check should be made to see if more transmissions are required, the details of which may be held in the "store" described in the above paragraph.

When called the ISR should check the Interrupt Status Register. If the PTX bit is set then this indicates that the packet

was transferred without error and the transmission can be flagged as successful. If the TXE bit is set, then an error has occurred and more information is required. This extra information is provided by the Transmit Status Register (TSR) from which the relevant error flags and statistics counters can be updated before the transmission can be flagged as completed with error. If collisions occurred during transmission then the COL bit of the TSR is set and the number of collisions that occurred is held in the Number of Collisions Register (NCR). The software can use this register to keep statistics on the amount of collisions occurring on a network.

### 6.0 RECEPTION SEQUENCE

#### 6.1 The Buffer Ring

As packets are received they are placed in to the buffer ring and as they are processed they are removed from the ring. At initialization an area of memory is allocated to act as the receive buffer ring and the AT/LANTIC's buffer management scheme controls the operation of the memory.

Four pointers are used to control the ring; the page start (PSTART) and page stop (PSTOP) pointers to determine the size of the buffer ring, the current page (CURR) pointer to determine where the next packet will be written to from the network and the boundary (BNDRY) pointer to show where the next packet to be read and processed lies. As packets are received, the boundary pointer follows the current page pointer around the ring. During operation the page start and page stop registers remain unchanged.

The receive buffer ring is divided into 256 byte buffers (called "pages") and these are linked together as required by the received packets. Since all AT/LANTIC registers are byte wide the ring pointers refer to the page (256 byte) boundaries, see *Figure 10*.

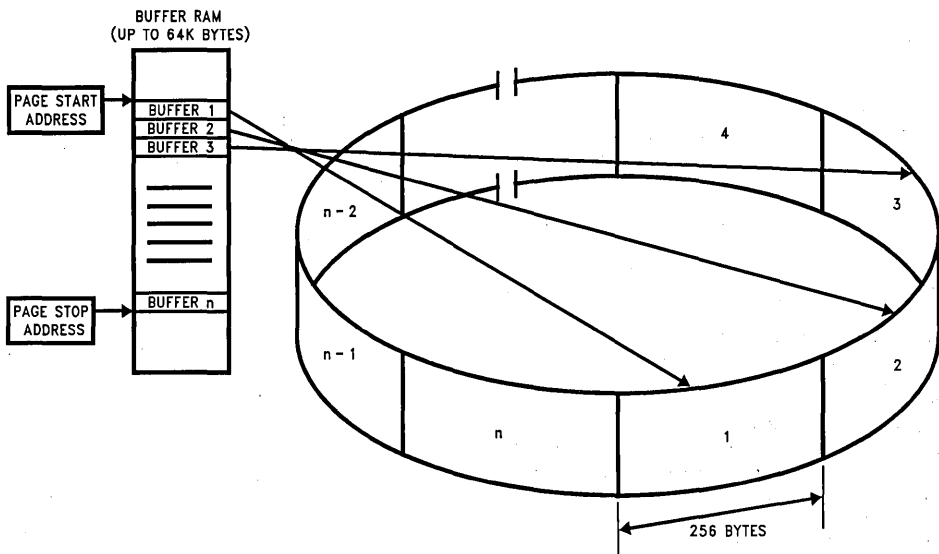


FIGURE 10. The Receive Buffer Ring

TL/F/11820-3

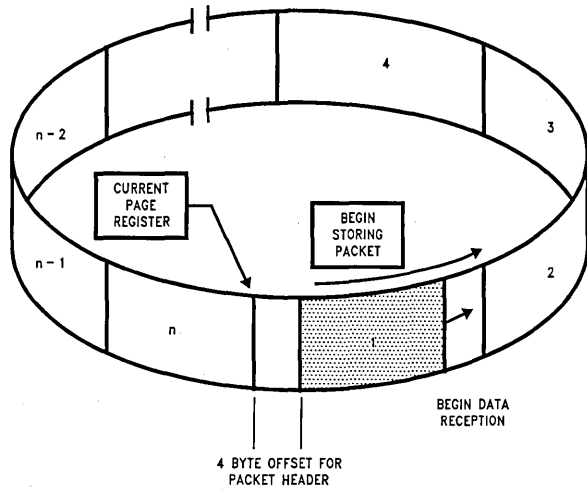
On a valid reception the packet is placed in the ring at the page pointed to by CURR plus a 4 byte offset (see *Figure 11*). The packet is transferred to the ring through the AT/LANTIC which links the page buffers as necessary until the complete packet is received. The first and last buffers (PSTART and PSTOP) are linked just as the first and second buffers would be. At the end of a reception, the status from the Receive Status Register (RSR), a pointer to the next packet and the byte count of the current packet are written into the 4 byte offset (see *Figure 12*).

**6.2 Removing a Packet**

Once packets are in the receive buffer ring they must be processed. The AT/LANTIC supports two differing adapter

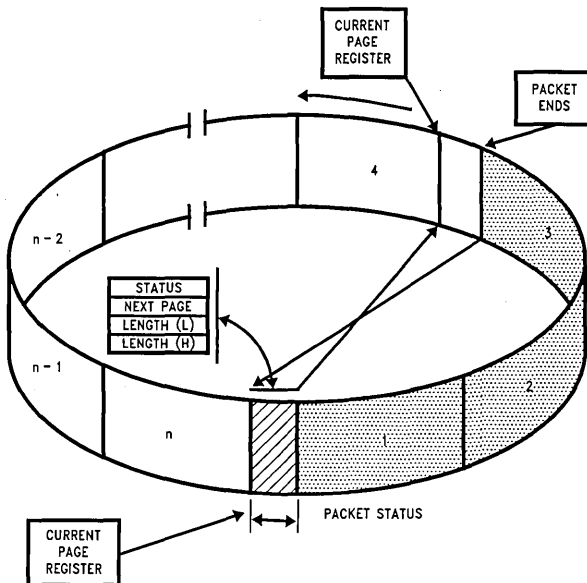
architectures which have their own method of accessing the buffer memory, these are discussed in Section 4.0 of this document. As packets are removed from the buffer ring, the boundary (BNDRY) pointer must be updated. The BNDRY follows CURR around the ring (see *Figure 13*).

If the current local DMA address (the place where a newly received packet is being stored) ever reaches BNDRY then the ring is full and any more receptions cannot be achieved until some processing has been done on the ring. This condition is known as overflow. More details on this condition and how to overcome it are given in Section 6.3.



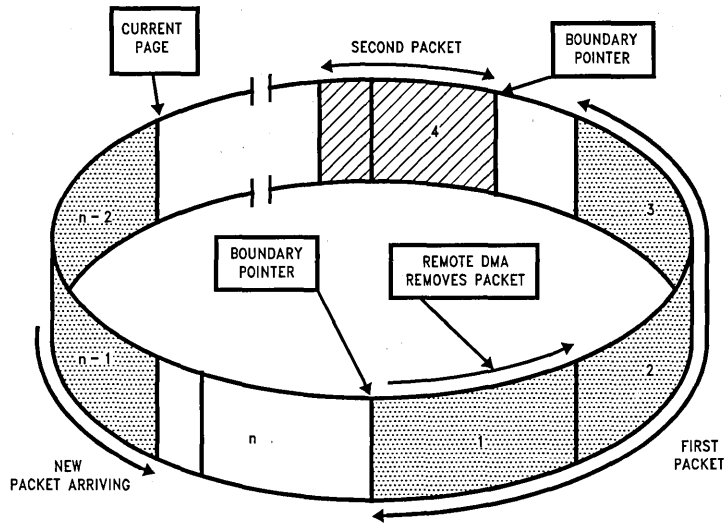
**FIGURE 11. Receiving a Packet**

TL/F/11820-4



**FIGURE 12. Receive Packet Header**

TL/F/11820-5



TL/F/11820-6

FIGURE 13. Removing a Packet

When the CURR and BNDRY pointers are equal, the buffer ring can either be full or empty. To ensure that the software never misinterprets this condition the BNDRY pointer can be kept one less than the CURR pointer when the ring is empty, and only equal to CURR when the ring is full, as shown below.

1. Use a variable (Next\_pkt) to indicate where the next packet will be removed from the buffer ring.

2. At initialization set (see Section 3.3):

```
BNDRY = PSTART
```

```
CURR = PSTART + 1
```

```
Next_pkt = PSTART + 1
```

3. After each packet is removed from the ring, use the next packet pointer in the header information (the second byte of the header), HNXTPKT and set:

```
Next_pkt = HNXTPKT
```

```
BNDRY = HNXTPKT
```

```
if (BNDRY < PSTART)
```

```
  BNDRY = PSTOP - 1
```

### THE SEND PACKET COMMAND

When in I/O mode the Remote DMA channel can be automatically initialized to transfer a single packet from the receive buffer ring. The transfer is initiated by issuing a "send packet" command, setting bits RD1, RD0 and STA in the command register. The DMA will be initialized to the value of the BNDRY pointer and the Remote Byte Count registers will be initialized to the values found in the buffer header of each received packet. After the data has been transferred, the BNDRY pointer is advanced and the Remote DMA is then prepared to receive the next packet.

This method does not require the manual updating of registers as discussed in Steps 1-3, however it does limit the software to accessing a packet just once.

The receive sequence is initiated by an interrupt generated by the AT/LANTIC when data is ready to be removed from the ring or an overflow has occurred. The Interrupt Service Routine should then interrogate the Interrupt Status Register which flags a PRX bit when a packet is ready to be removed or OVW if the buffer ring is full. When this PRX bit is set then a receive subroutine following the above sequence should be called, if the OVW bit is set the overflow routine discussed in Section 6.3 should be called.

The receive buffer "ring" is a linear section of RAM forced into a ring by linking the end (PSTOP) and beginning (PSTART) of the buffer. Some of the received packets shall overlap or "wraparound" this link in the buffer, i.e. the start of the packet is held up to PSTOP and the rest is held from PSTART. This condition is automatically dealt with when in I/O mode however Shared Memory mode requires more care. In Shared Memory mode a "wraparound" packet has to be transferred in its two sections, i.e., the section up to PSTOP and the section from PSTART.

A pseudo code example of the sequence is given below.

### RECEIVE ROUTINE

```
Receive_subroutine()
```

```
{
// This routine loops until all the packets
// currently held in the buffer ring are
// removed.
  while(Next_pkt != CURR)
  {
```

```

// Get the 4 byte header from the packet
// pointed to by Next_pkt, the method of
// removing the data depends on the mode of
// the AT/LANTIC.
    status = read_status();
// If in shared memory mode then check if
// the packet wraps around the PSTOP
// pointer.
    if(in_shared_memory_mode)
    {
        if (status.length + Next_pkt > PSTOP)
        {
            transfer_up_to_PSTOPO;
            transfer_from_PSTART();
        }
        else
            transfer_all_at_once();
    }
    else
// I/O mode automatically carries out
// wraparound.
        transfer_all_once();
// Set the Next_pkt pointer to equal the
// next packet pointer in the status bytes.
    Next_pkt = status.next_packet;
// Update the boundary pointer.
    BNDRY = Next_pkt - 1;
    if(BNDRY < PSTART)
        BNDRY = PSTOP - 1;

    WRITE(BNDRY);
} // end of while loop.
}

```

### 6.3 Dealing with Overflows

In heavily loaded networks it is possible for the receive buffer ring to be filled with packets that still require processing, i.e., the CURR pointer reaches the BNDRY pointer. If this situation occurs then the AT/LANTIC suspends further receptions and posts an overflow (OVW) interrupt.

In the event of an overflow condition occurring it is necessary to follow the routine given below. If this routine is not adhered to then the AT/LANTIC may act in an unpredictable manner. A flow chart of the routine is given in *Figure 8*.

**Note:** It is necessary to define a variable in the driver, which will be called "Resend".

1. Read and store the value of the TXP bit in the AT/LANTIC Controller's Command Register.
2. Issue the STOP command to the AT/LANTIC Controller. This is accomplished by setting the STP and RD2 bits in the AT/LANTIC Controller's Command Register.
3. Wait for at least 1.6 ms. Since the AT/LANTIC Controller will complete any transmission or reception that is in

progress, it is necessary to time out for the maximum possible duration of an Ethernet transmission or reception. By waiting 1.6 ms this is achieved with some guard band added. Previously, it was recommend that the RST bit of the Interrupt Status Register be polled to insure that the pending transmission or reception is completed. This bit is not a reliable indicator and subsequently should be ignored.

4. Clear the AT/LANTIC Controller's Remote Byte Count registers (RBCRO and RBCR1).
5. Read the stored value of the TXP bit from step 1, above.

If this value is a 0, set the "Resend" variable to a 0 and jump to step 6.

If this value is a 1, read the AT/LANTIC Controller's Interrupt Status Register. If either the Packet Transmitted bit (PTX) or Transmit Error bit (TXE) is set to a 1, set the "Resend" variable to a 0 and jump to step 6. If neither of these bits is set, place a 1 in the "Resend" variable and jump to step 6.

This step determines if there was a transmission in progress when the stop command was issued in step 2. If there was a transmission in progress, the AT/LANTIC Controller's ISR is read to determine whether or not the packet was transmitted by the AT/LANTIC Controller. If neither the PTX nor TXE bit was set, then the packet will essentially be lost. If a packet was "lost" then a transmit command can be reissued to the AT/LANTIC Controller once the overflow routine is completed (as in step 11). Also, it is possible for the AT/LANTIC Controller to deter indefinitely, when it is stopped on a busy network. Step 5 also alleviates this problem. Step 5 is essential and should not be omitted from the overflow routine, in order for the AT/LANTIC Controller to operate correctly.

6. Place the AT/LANTIC Controller in either mode 1 or mode 2 loopback. This can be accomplished by setting bits D2 and D1, of the Transmit Configuration Register, to "0,1" or "1,0", respectively. Further explanation of loopback can be found in the AT/LANTIC data sheet under Section 6.5.
7. Issue the START command to the AT/LANTIC Controller. This can be accomplished by setting the START and RD2 bits in the Command Register. This is necessary to activate the AT/LANTIC Controller's Remote DMA channel.
8. Remove one or more packets from the receive buffer ring.
9. Reset the overwrite warning (OVW, overflow) bit in the Interrupt Status Register.
10. Take the AT/LANTIC Controller out of loopback. This is done by writing the Transmit Configuration Register with the value it contains during normal operation. (Bits D2 and D1 should both be programmed to 0.)
11. If the "Resend" variable is set to a 1, reset the "Resend" variable and reissue the transmit command. This is done by the TXP bit in the Command Register. If the "Resend" variable is 0, nothing needs to be done.

**Note:** If Remote DMA is not being used, the AT/LANTIC Controller does not need to be started before packets can be removed from the receive buffer ring. Hence, step 8 could be done before step 7, eliminating or reducing the time spent polling in step 5.

**Note:** When the AT/LANTIC Controller is in STOP mode, the Missed Packet Tally counter is disabled.

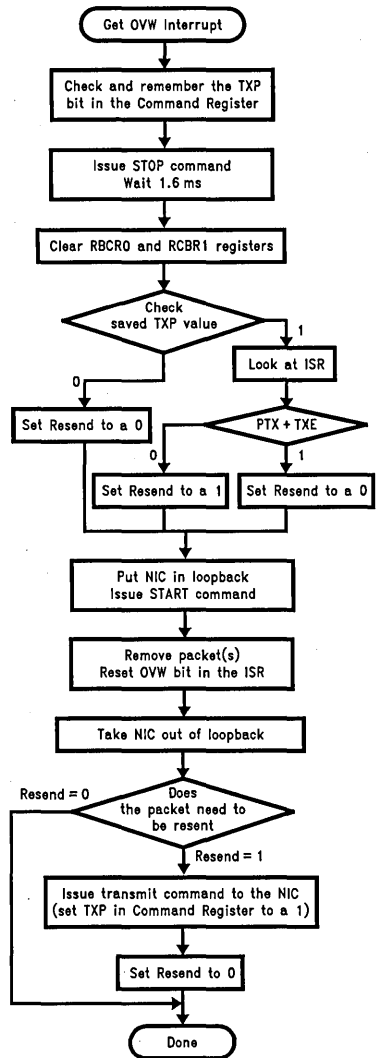


FIGURE 14. Overflow Routine

TL/F/11820-7

# I/O Channel Ready Considerations for the DP83902EB-AT

National Semiconductor  
Application Note 892



## INTRODUCTION

Some PC-AT® compatible systems that use bus interface chip sets with modified timing characteristics such as some Chips & Technologies or VLSI technologies chip sets have different timing requirements that require modification to 16-bit I/O mapped designs to operate properly. 8-bit I/O mapped designs do not require this modification.

This paper describes the timing issues associated with the DP83902EB-AT and methods of fixing these timing incompatibilities.

## OVERVIEW

The timing inconsistency is the time of assertion and deassertion of the IOCHRDY bus signal. IOCHRDY floats active (ready) when not driven by an I/O card. Normally, an I/O card should drive IOCHRDY low (not ready) to insert wait states only after the address and I/O read or write signal are asserted. However, on some PC-AT buses, during a 16-bit I/O operation the bus controller actually samples the IOCHRDY signal before the I/O read or write signal is asserted. To correct the early sampling of IOCHRDY by the bus controller, the I/O card can drive IOCHRDY based only on an address decode, thus allowing IOCHRDY to be asserted earlier and in the proper state when it is sampled by the bus controller.

Remote read cycles are executed by National Semiconductor's DP83902 ST-NIC™, Network Interface Controller with integrated twisted pair, during packet reception. A remote read cycle will be used to illustrate how the early sampling of IOCHRDY is compensated. Also, the PAL® equations needed to generate IOCHRDY and the necessary modifications to account for the early sampling of IOCHRDY for the DP83902EB-AT will be described.

```
NRDYEN = !( !NSTNICB & !NIOR & !SA4 & NACK
            # !NSTNICB & !NIOW & !SA4 & NACK
            # !NSTNICB & !PRQ & !NIOR & SA4 & !SA3
            # !NSTNICB & !PRQ & !NIOW & SA4 & !SA3 );
enable NIOCHRDY = !NRDYEN;
```

```
NIOCHRDY = (0);
```

FIGURE 1. "Normal" IOCHRDY PAL Equations for DP83902EB-AT

## REMOTE READ CYCLE

The ST-NIC receives data from the network and transfers the data to the local buffer memory using the local DMA channel. The ST-NIC then executes a remote DMA read to transfer the received data from the buffer memory to the host memory through the I/O port latch.

Referring to *Figure 2*, the remote read cycle functions as follows:

1. The ST-NIC reads a word from local buffer memory asserting  $\overline{MRD}$  and writes the word into the I/O port latch asserting  $\overline{PWR}$ .
2. The ST-NIC asserts the request line (PRQ) to inform the host a word is in the I/O port latch.
3. The host reads the I/O port asserting  $\overline{IOR}$ . IOCHRDY is asserted if  $\overline{IOR}$  occurs before PRQ to extend the assertion time of  $\overline{IOR}$ , effectively wait-stating the host so that the host-I/O handshake can occur.
4.  $\overline{RACK}$  is asserted to signal to the ST-NIC that the host has read the word from the I/O port latch.

Steps 1–4 are repeated until all words are transferred.

## "NORMAL" IOCHRDY TIMING DURING A REMOTE READ CYCLE

The "normal" IOCHRDY PAL equations are shown in *Figure 1* and the complete PAL equations for U2 in the DP83902EB-AT are shown in AN-752 in the 1992 Local Area Network Databook. The corresponding timing diagram is shown in *Figure 2*. Referring to *Figure 1* and AN-752, whenever NRDYEN is true, IOCHRDY is driven, otherwise it will be held TRI-STATE®. The first two terms in the NRDYEN function are for a slave read and write, respectively. The next two terms are for remote read and write, respectively. There are 7 unique variables in the function NRDYEN. Namely, NSTNICB, NIOR, NIOW, NACK, PRQ, SA4 and SA3. A brief explanation of each follows.

NSTNICB—(SA9–SA5) chip select the I/O card. This is the base address of the I/O card.

NIOR—( $\overline{\text{IOR}}$ ) the I/O read command signal.

NIO $\overline{\text{W}}$ —( $\overline{\text{IOW}}$ ) the I/O write command signal.

NACK—( $\overline{\text{ACK}}$ ) Used during slave read/write cycles to indicate a successful register (byte) transfer.

PRQ—Used during remote read/write cycles to inform the host that the NIC has written a word in the I/O port latch.

SA4 and SA3—Address signals from the I/O bus. The states of these signals determine if the address on the I/O bus is for the I/O port latch or the NIC registers.

The on board IOCHR $\overline{\text{D}}$ Y signal will only be driven to wait state the host until the NIC writes to the I/O port latch and asserts PRQ.

The remote word transfer is complete after the  $\overline{\text{RACK}}$  signal is asserted indicating the host has read the word from the I/O port latch.

**MODIFIED IOCHR $\overline{\text{D}}$ Y TIMING DURING A REMOTE READ CYCLE**

The modified IOCHR $\overline{\text{D}}$ Y PAL equations shown in *Figure 4* implement both the "normal" IOCHR $\overline{\text{D}}$ Y timing and the modified IOCHR $\overline{\text{D}}$ Y timing. These equations replace the section of the U2 PAL equations in AN-752 shown in *Figure 1*. The modified IOCHR $\overline{\text{D}}$ Y schematic and timing diagram are shown in *Figures 3* and *5* respectively.

NIOCHR and NIOCHR.OE (enable NIOCHR) together are for slave read and remote read. Also, NIOCHW and NIOCHW.OE (enable NIOCHW) together are for slave write and remote write. These signals (NIOCHR and NIOCHW) together produce the "normal" IOCHR $\overline{\text{D}}$ Y signal.

NIOCHC and NIOCHC.OE (enable NIOCHC) together modify the IOCHR $\overline{\text{D}}$ Y timing. In the modified IOCHR $\overline{\text{D}}$ Y timing, IOCHR $\overline{\text{D}}$ Y is driven when the I/O card address is decoded, thus driving IOCHR $\overline{\text{D}}$ Y before it is sampled by the host.

By driving IOCHR $\overline{\text{D}}$ Y on an address decode the possibility exists that it may be driven on a memory access instead of an I/O access. IOCHR $\overline{\text{D}}$ Y must be held TRI-STATE if the host issues a memory access command signal (MEMR or MEMW) because if these signals become active after an address decode to the I/O card this means that the address was a memory address and not an I/O address. Thus, the address was not intended for the I/O card and driving IOCHR $\overline{\text{D}}$ Y may cause contention with the memory.

PRQ is held TRI-STATE until the ST-NIC's Data Configuration Register (DCR) is programmed. A pull-up resistor is needed to guarantee PRQ is asserted while its TRI-STATE to prevent IOCHR $\overline{\text{D}}$ Y from "getting stuck" low and effectively locking up the host.

Driving IOCHR $\overline{\text{D}}$ Y early must be an option since this has been seen to cause problems on some PC's, but fixes problems on others. The variable CLONEN is a jumper used to switch the IOCHR $\overline{\text{D}}$ Y signal characteristics to "normal" or modified timing. The inverted SYSCLK (System Clock) and BALE (Bus Address Latch Enable) signals are AT bus signals and are used to assert CLONEN when the bus address is valid, which is indicated by BALE. CLONEN is asserted when BALE is asserted and deasserted after BALE is deasserted and the next SYSCLK rising edge occurs, allowing enough time for the I/O command signal to occur. At this point, the modified IOCHR $\overline{\text{D}}$ Y timing is TRI-STATE and the "normal" IOCHR $\overline{\text{D}}$ Y timing is enabled.

EN16 is used to determine if the data transfer is 8 or 16 bits. The modified IOCHR $\overline{\text{D}}$ Y is necessary only during 16-bit I/O accesses because the early sampling of IOCHR $\overline{\text{D}}$ Y only occurs during 16-bit I/O accesses.

To incorporate the IOCHR $\overline{\text{D}}$ Y modification in the DP83902EB-AT, replace the section of the U2 PAL equations shown in *Figure 1* with the PAL equations of *Figure 4* and add a 74F74 D flip-flop and a jumper to the evaluation board to hard-wire CLONEN to U2.

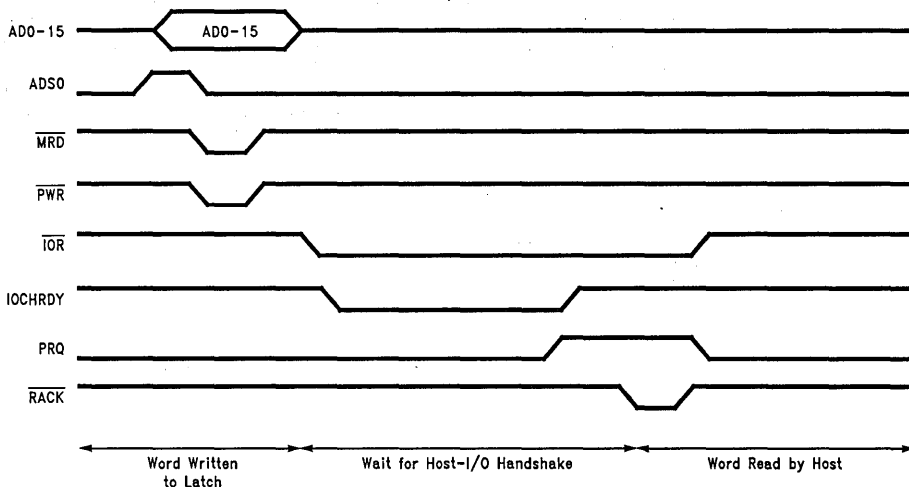
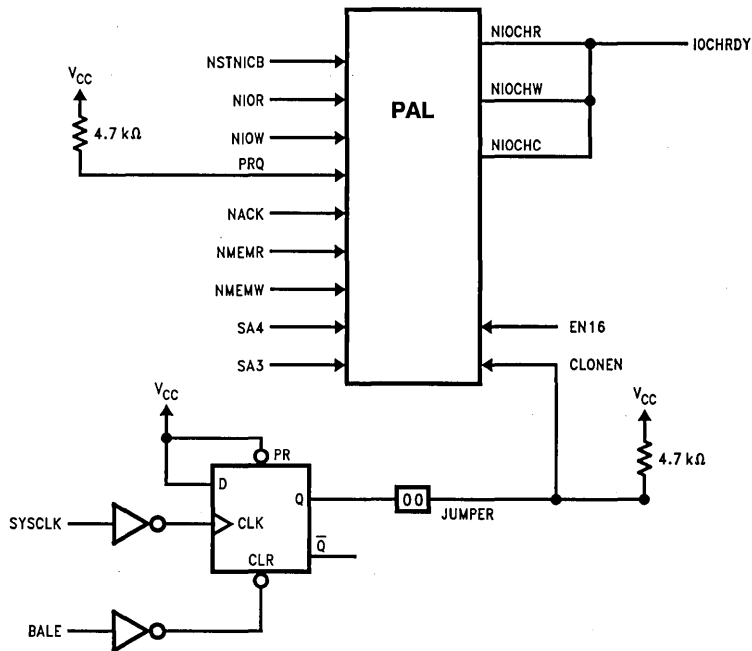


FIGURE 2. "Normal" Remote Read Cycle for DP83902EB-AT

TL/F/11827-1





TL/F/11827-2

FIGURE 3. Schematic of IOCHRDI Modification Showing Signals Required for IOCHRDI Change

```
NIOCHR = !(IPRQ & SA4 & !SA3 # NACK & !SA4);
enable NIOCHR = !(INSTNICB & !NIOR);
```

```
NIOCHW = !(IPRQ & SA4 & !SA3 # NACK & !SA4);
enable NIOCHW = !(INSTNICB & !NIOW);
```

```
NIOCHC = (0);
enable NIOCHC = !(INSTNICB & NMEMR & NMEMW & !PRQ & !EN16 & !CLONEN & SA4 & !SA3);
```

FIGURE 4. Modified PAL Equations for DP83902EB-AT

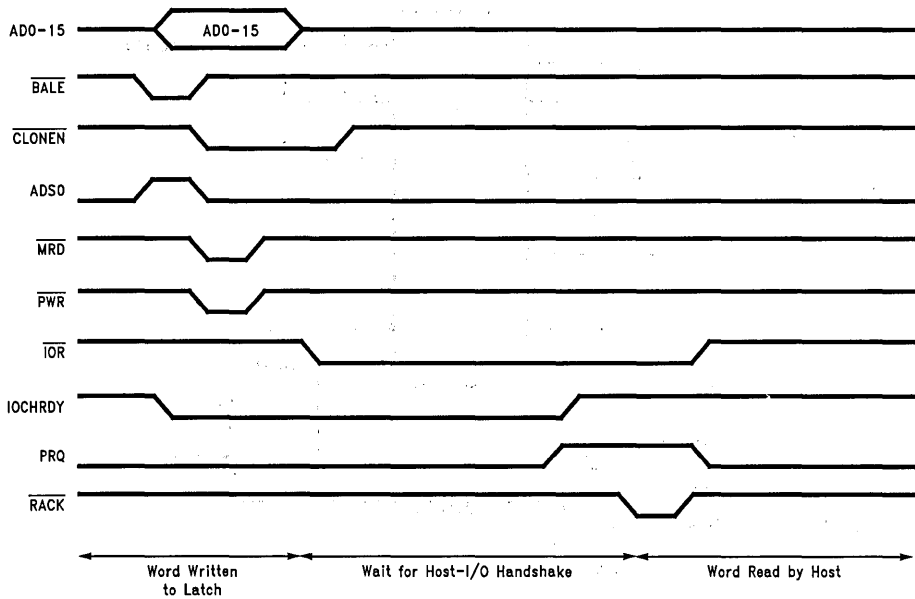


FIGURE 5. Modified Remote Read Cycle for DP83902EB-AT

TL/F/11827-3

# DP83905EB-AT AT/LANTIC™ Hardware Users' Guide

National Semiconductor  
Application Note 897  
Larry Wakeman



AN-897

## TABLE OF CONTENTS

### 1.0 DP83905 ETHERNET INTERFACE DESIGN

- 1.1 Design Decisions and Implementation Details
  - 1.1.1 ISA BusConnection
  - 1.1.2 Interrupts
  - 1.1.3 Buffer RAM
  - 1.1.4 Crystal/Oscillators
  - 1.1.5 Boot ROM (Remote Program Load PROM)
  - 1.1.6 Cable Interfaces
  - 1.1.7 LED Interface Options
  - 1.1.8 Low Power
  - 1.1.9 Configuration
- 1.2 Layout Considerations
  - 1.2.1 ISA Bus
  - 1.2.2 Twisted Pair (TPI)
  - 1.2.3 Attachment Unit Interface (AUI)
  - 1.2.4 Thin Ethernet (Coax)
  - 1.2.5 PLL
- 1.3 EEPROM Programming
- 1.4 Additional End-User Requirements
  - 1.4.1 Configuration Software
  - 1.4.2 Driver Software
  - 1.4.3 Documentation (Installation Guide)

### 2.0 REFERENCE INFORMATION

- 2.1 Architecture
  - 2.1.1 NIC Core
  - 2.1.2 NE2000
  - 2.1.3 Shared Memory Mode
- 2.2 Memory and IO maps
  - 2.2.1 NE2000 8 Bit and 16 Bit
  - 2.2.2 Shared Memory Mode

- 2.3 ISA Bus Description
  - 2.3.1 I/O Cycles
  - 2.3.2 Memory Cycles
  - 2.3.3 Cycle Timing
  - 2.3.4 8- and 16-Bit Cycles
  - 2.3.5 DMA and Refresh Cycles
  - 2.3.6 Bus Timing Compatibility Modes
- 2.4 Boot PROM
- 2.5 Boot PROM and RAM Timing Calculations
  - 2.5.1 Boot PROM/RAM ISA Read Timing
  - 2.5.2 Boot PROM/RAM ISA Write Timing
  - 2.5.3 RAM DMA Timing
- 2.6 Fast Read Feature
- 2.7 Reset Operation

### INTRODUCTION

The AT/LANTIC Ethernet controller provides a simple method of interfacing a PC ISA (Industry Standard Architecture) bus based system to Ethernet. The AT/LANTIC controller emulates a popular adapter card for PC compatibles: the Novell® NE2000 *Plus* card. The AT/LANTIC also implements a shared memory mode that provides added flexibility and performance. The high level of integration ensures a small and cost-effective solution. In order to use a network interface, driver software is required that is specific to the network interface and the network operating system. Since the AT/LANTIC is compatible with the most popular adapter cards, driver support is second to none.

### ABOUT THIS GUIDE

This guide is written for hardware design engineers wishing to develop an Ethernet interface using AT/LANTIC. The guide is written in two sections. The first part is a step-by-step examination of the design process. The second section provides reference material with additional detailed information.

**1.0 DP83905 ETHERNET INTERFACE DESIGN**

**1.1 Design Decisions and Implementation Details**

This section of the AT/LANTIC guide is intended to ask the designer to choose which options are best for each application, whether it is an adapter card, a motherboard or some other embedded design. (Note that Application Note AN-844 "DP83905EB-AT AT/LANTIC™ Demonstration Board" shows detailed schematics of an AT/LANTIC solution.)

**1.1.1 ISA Bus Connection**

The AT/LANTIC supports both 8- and 16-bit ISA bus configurations. An 8-bit interface is cheaper than 16-bit (less board space and only one SRAM), but has lower performance. There are three options for connecting the bus interface as described:

1. For an 8-bit interface, pins SD8-15, SBHE, LA17-23, MWR, MRD, M16, and I016 should be left unconnected, and DWID should be tied low.
2. For a 16-bit interface, DWID should be tied high.
3. For a 16-bit adapter card that can be used in 8- and 16-bit slots, DWID should be connected to pin D29 of the ISA bus. In a 16 bit slot this is +5V. In an 8-bit slot, this is unconnected, and a pull-down within AT/LANTIC will ensure that 8-bit mode is selected.

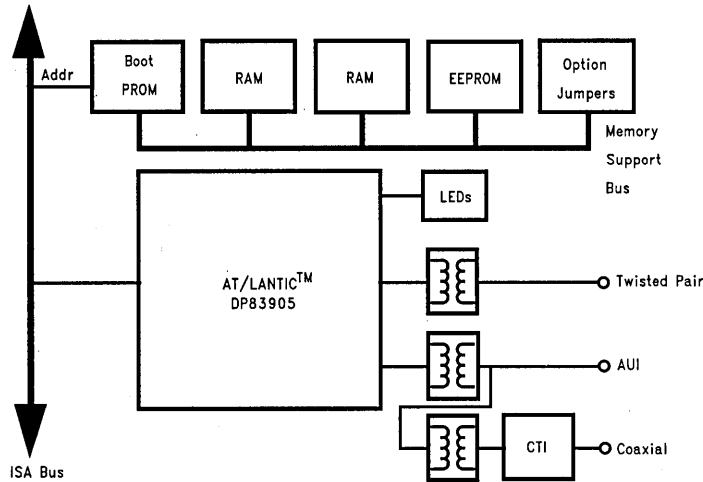
**1.1.2. Interrupts**

The AT/LANTIC has 4 dedicated interrupt output pins. This allows the user to select the required interrupt without changing jumpers on the board. In some applications, the choice of 4 interrupts may not be sufficient. The AT/LANTIC allows you to use an external decoder to expand the choice to 8 interrupts.

For 4 interrupts, connect each of the pins INT0-3 directly to the interrupt signals. We strongly recommend using the following scheme to promote software compatibility across AT/LANTIC platforms:

AT/LANTIC	PC
INT0	IRQ3
INT1	IRQ4
INT2	IRQ5
INT3	IRQ9

Configuration register C bit 5 "INTMOD" should be set to "0" (see Section 1.1.9).



**FIGURE 1. Simplified Block Diagram of AT/LANTIC Application**

TL/F/11850-1

For 8 interrupts, an external decoder is required. This decoder must have the following properties:

INT0-2 selects which 1 of the 8 outputs is driven. This output follows the logic level of INT3. The 7 unselected outputs must remain high impedance.

The decoder outputs are connected to 8 ISA interrupts. We recommend using the following scheme to promote software compatibility across AT/LANTIC platforms:

AT/LANTIC			PC
INT2	INT1	INT0	
0	0	0	IRQ3
0	0	1	IRQ4
0	1	0	IRQ5
0	1	1	IRQ9
1	0	0	IRQ10
1	0	1	IRQ11
1	1	0	IRQ12
1	1	1	IRQ15

Configuration register C bit 5 "INTMOD" should be set to "1" (see Section 1.1.9). The 8 interrupt mode interface can be implemented in a single GAL as shown in Figure 2.

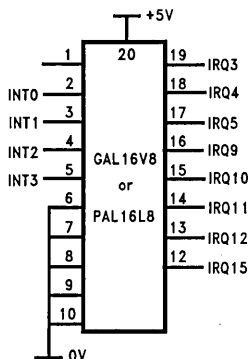


FIGURE 2. GAL/PAL Circuit Diagram

The PAL or GAL equations should be:

- IRQ3 = INT3;
- IRQ4 = INT3;
- IRQ5 = INT3;
- IRQ9 = INT3;
- IRQ10 = INT3;
- IRQ11 = INT3;
- IRQ12 = INT3;
- IRQ15 = INT3;
- IRQ3.oe = /INT2 & /INT1 & /INT0;
- IRQ4.oe = /INT2 & /INT1 & INT0;
- IRQ5.oe = /INT2 & INT1 & /INT0;
- IRQ9.oe = /INT2 & INT1 & INT0;
- IRQ10.oe = INT2 & /INT1 & /INT0;
- IRQ11.oe = INT2 & /INT1 & INT0;
- IRQ12.oe = INT2 & INT1 & /INT0;
- IRQ15.oe = INT2 & INT1 & INT0;

### 1.1.3. Buffer RAM

The standard RAM sizes for 8- and 16-bit cards are 8k and 16k respectively, using one or two 8k x 8 RAMs. The AT/LANTIC will also support the use of 32k x 8 RAMs giving 4 times the standard RAM capacity. Using additional memory can provide a performance boost in server applications.

Note that the use of 32k x 8 RAMs will make the design INCOMPATIBLE with NE2000.

For 8k x 8 RAMS, MSA1-13 are used. Configuration register C bit 4 "COMP" should be set to "0".

For 32k x 8 RAMS, MSA1-15 are used. Configuration register C bit 4 "COMP" should be set to "1".

When making an 8-bit-only interface, only one RAM is required, connected to MSD0-7. For a standard ISA bus configuration, 100 ns RAMs, or faster, should be used. Details of how to calculate RAM speeds are given in Section 2.5.

### 1.1.4 Crystal/Oscillators

#### 1.1.4.1 20 MHz Clock

AT/LANTIC requires a 20 MHz ±0.01% clock. Both crystal and oscillator solutions are supported. In general, a crystal solution is cheaper, however if a suitable clock is already available on your board, it could be used.

To use a crystal, use the following circuit.

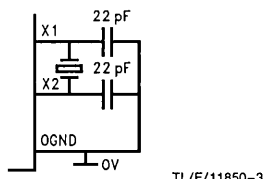


FIGURE 3. Crystal Circuit

The value of the capacitors should be 26 pF minus the printed circuit board trace capacitance. (Typical trace and pin capacitance is about 4 pF).

1. Note that the following rules should be applied:
2. The signal traces should be short.
3. The ground return path to the capacitors should be short, and should connect to the AT/LANTIC OGND pin.
4. There should not be other signal traces through the region of the crystal.
5. It may help to place a ground plane under the crystal.
6. The crystal should conform to the following specifications:

- AT cut parallel resonant crystal
- Series resistance ≤ 25Ω
- Specified load capacitance ≤ 20 pF
- Accuracy 0.005% (50 ppm)
- Typical load 50 μW-75 μW

7. Do not connect X1 or X2 to anything else.

Alternately, an oscillator may be used. If used the oscillator output should be connected directly to X1. X2 should be left unconnected. The clock should be better than 40:60 duty cycle, and should have a good quality waveform.

#### 1.1.4.2 BSCLK

The AT/LANTIC core can be clocked either by the 20 MHz clock, or by a separate clock input "BSCLK". The maximum clock rate is 20 MHz, so the 20 MHz clock is usually used.

To use the 20 MHz clock, connect the BSCLK pin to 0V, and set Configuration register C bit 6 "CLKSEL" to "0". To use another clock source, connect the clock source to the BSCLK pin, and set Configuration register C bit 6 "CLKSEL" to "1". See Section 1.1.9 for more information about configuration data.

**1.1.5 Boot ROM (Remote Program Load PROM)**

Depending on the intended use of your design, you may wish to provide a boot PROM, or a socket for a boot PROM. A boot PROM allows the PC to load the operating system from a server on the network, without needing a disk drive on the PC. AT/LANTIC supports boot PROM sizes of 8k, 16k, 32k and 64k. The AT/LANTIC also supports the use of FLASH ROM through the use of a write signal. This may be used to allow in-situ programming, or updating of boot PROM code by the user.

Adaptor cards are typically supplied with an empty boot PROM socket. Boot PROMs are available from network operating system vendors or third parties. They are typically 8 kbytes or 16 kbytes in a 28-pin package. The boot prom size can be configured by the user to be any size and at any address between C0000h and DFFFFh, using configuration register C bits 0-3 "BPS0-3".

Configuration register B bit 6 "BPWR" specifies if the AT/LANTIC will allow write access to the PROM: "0" = read only, "1" = write enabled. (See Section 1.1.9, Configuration.)

For a standard ISA bus configuration, 250 ns PROM, or faster, should be used. Details of how to calculate boot PROM speeds are given in Section 2.5.

**1.1.6 Cable Interfaces**

AT/LANTIC supports three types of cable interface:

- TPI 10Base-T, Twisted Pair
  - AUI for connection to 10Base5, also known as "thick" Ethernet
  - Coax 10Base2, also known as thin Ethernet
- The TPI interface connects to the AT/LANTIC as follows:

The AT/LANTIC selects between the AUI/coax interfaces and TPI internally. Note that, in addition to 10Base-T compatible twisted pair, the AT/LANTIC supports the use of higher loss cable systems. This allows cables to be longer than the 10Base-T specification, or allows the use of shielded cables. Refer to Section 4.8 of the AT/LANTIC data sheet.

The AUI interface requires an isolation transformer, and resistors, as shown in Figure 5. The AUI interface enables the use of an off board transceiver to connect to other types of media such as Thick Ethernet or Fiber Optic cabling.

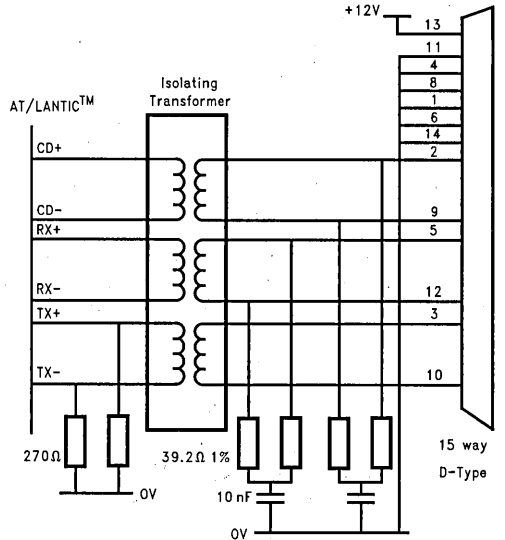


FIGURE 5. AUI Interface Circuit

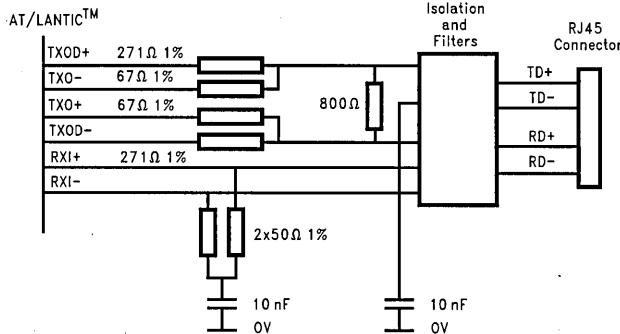


FIGURE 4. Twisted Pair Interface

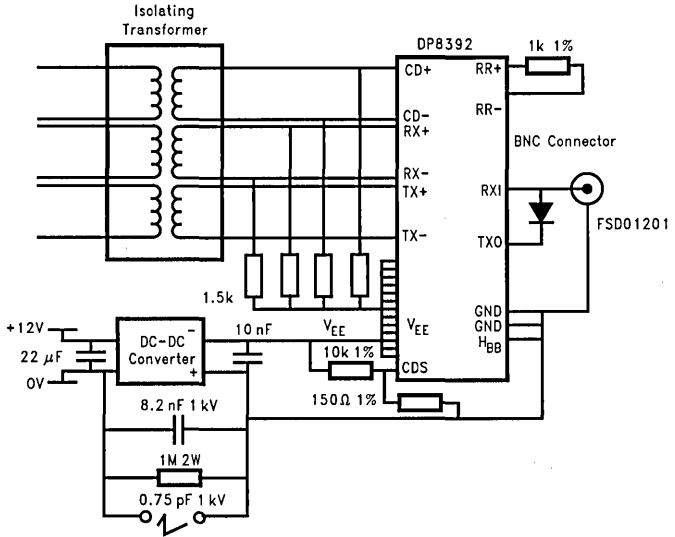


FIGURE 6. Coax Interface Circuit

TL/F/11850-6

The Coax interface requires an isolation transformer, a DC-DC converter and the DP8392 Coax Transceiver Interface as shown in *Figure 6*.

If it is desirable to implement a design that supports both AUI and Thin Ethernet cabling schemes, and provides for selection between the two interfaces the coax interface should connect to the AUI interface as shown in *Figure 7*. In order to use the AUI connector, the coax interface must be disabled. AT/LANTIC™ selects AUI or coax with an output pin, called "THIN". This signal is used to switch-on the DC-DC converter of the coax interface when THIN is high. When coax is selected, an AUI cable must not be connected to the AUI connector.

*Figure 7* shows a DC-DC converter with an enable input. A DC-DC converter without an enable input can be used, if the supply to the converter is switched by an FET.

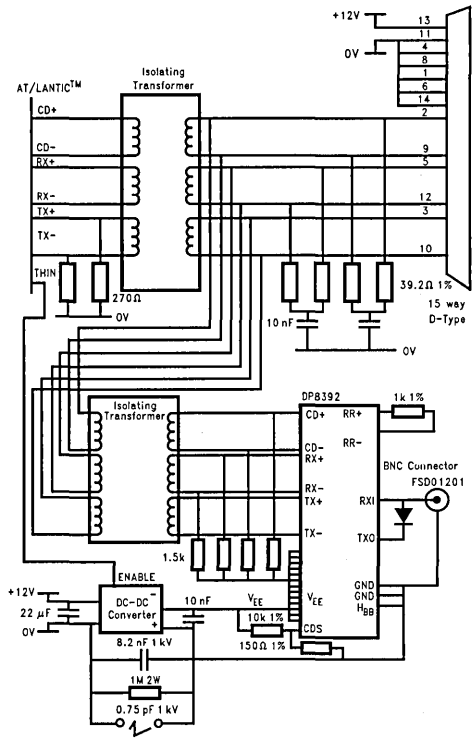
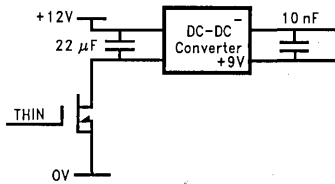


FIGURE 7. AUI and Coax Interface Circuit

TL/F/11850-7



TL/F/11850-8

**FIGURE 8. FET Switch for DC-DC Converter**

The cable interfaces utilize a number of magnetic components as shown in the previous figures. The specifications for these components are shown below.

AUI and Coax Interface isolation transformer:

- 1:1 turns ratio
- 100  $\mu$ H inductance
- 500V isolation

Coax DC-DC converter:

- +12V input (could use +5V connected to the +5V supply)
- 9V output voltage
- 200 mA output current
- 500V isolation

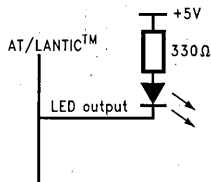
Active-high enable input (if required)

The 10BASE-T interface uses an integrated Filter-Transformer-Choke combination.

Specific component recommendations are listed in the Local Area Networks Data Book in the Magnetics Vendors Application Note.

### 1.1.7 LED Interface Options

AT/LANTIC has dedicated outputs for driving several LEDs. These outputs are capable of a maximum current per LED output of 16 mA. Do not decrease the resistor value below 330 $\Omega$ , or the AT/LANTIC maximum current will be exceeded.



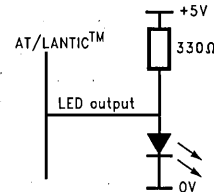
TL/F/11850-9

**FIGURE 9. LED Circuit**

It may be desirable to connect LEDs to any or all of the possible outputs. When used the LEDs can provide the following information:

- TxLED Transmit activity
- RxLED Receive activity (all packets on network)
- COLED Collision
- GDLNK TPI Link Ok, or testing disabled
- POLED TPI Polarity reversed

In some cases the polarity of the LED is reversed from that which the designer would like to use (e.g., the LED is off under a condition that the designer would like it to be on). To handle this, it is possible to use the following circuit, *Figure 10*, to invert the meaning of an LED (e.g., instead of a "Good Link" LED the LED can be defined as "Link fail"):



TL/F/11850-19

**FIGURE 10. Inverted LED Circuit**

### 1.1.8 Low Power

In some portable applications it is necessary to shut-off sub-systems in order to conserve power. AT/LANTIC has been designed so that it is possible to remove the power from the majority of the chip whilst still powering the ISA interface circuit, thus protecting the ISA bus. In low power state, the current drain is less than 100  $\mu$ A.

If "Low Power" is not required, tie the LOWPWR pin to 0V.

If "Low Power" operation is required, the following must be noted:

1. An external device is required to disconnect the PLLV<sub>CC</sub>, PV<sub>CC</sub>, OV<sub>CC</sub> and V<sub>CC</sub> pins. This device must have a low voltage drop when "on". A suitable device is the NDS9400 which has a 0.25 $\Omega$  on-resistance.
2. When the supply to the PLLV<sub>CC</sub>, PV<sub>CC</sub>, OV<sub>CC</sub> and V<sub>CC</sub> pins is turned off, the LOWPWR pin must be driven high. This disables internal buffers and reduces current drain to a minimum.
3. When recovering from low power state to normal operation, after restoring the power and driving LOWPWR low, the RESET pin must be driven high for more than 400  $\mu$ s to reset the controller.
4. Note that the DWID pin has an internal pull down. If this is tied high for 16-bit operation, it should be tied to the switched V<sub>CC</sub> to minimize current drain in the low power state.

### 1.1.9 Configuration

Three registers in the AT/LANTIC are loaded with configuration data at reset. This configuration specifies all the design choices (e.g., 4 or 8 interrupts) and the user installation choices (e.g., the I/O address).

This data can either be stored in the EEPROM, or be specified by resistors optionally connected to the CA0-7, CB0-7 and CC0-7 pins (also known as MSD0-15 and MSA1-8).

The benefits of the Jumperless EEPROM solution are that it is smaller and cheaper (no resistors or jumpers), and that the user can change the configuration without opening the computer. This may be especially important in a motherboard application. Additionally, the software used to change the configuration can attempt to check for address conflicts etc., and protect the user from making mistakes.



One potential problem with this solution, for adapter cards, is that the default settings as shipped may cause an address conflict in some cases. Such a conflict would prevent accesses to the card which are needed to change the address to a safe one.

To overcome this problem, AT/LANTIC has a "disabled" option. Configuration software can find an I/O address that is free, and then wake up the AT/LANTIC. Whilst disabled, AT/LANTIC monitors I/O accesses to 278h (a printer port). On the fourth consecutive write to 278h, AT/LANTIC loads the configuration from the write data, and wakes up.

The benefits of the Jumpered solution are that no software is needed to set the configuration. This may be important to system builders when choosing an adapter card, for whom jumpers may be faster to use than software.

Whichever configuration method you choose, you will need to determine, for each configuration bit, what the default value should be, and whether the bit can be changed by a user or is fixed by design. The following tables may assist you in this. Note that a full description of each configuration bit can be found in Section 5 of the AT/LANTIC data sheet.

Bit	Use
<b>Config A:</b>	
0	I/O Address
1	I/O Address
2	I/O Address
3	Interrupt
4	Interrupt
5	Interrupt (if 8 selected)
6	Fast read (See Section 1.1.9)
7	NE2000/Shared Memory
<b>Config B:</b>	
0	AUI/Coax/TPI
1	AUI/Coax/TPI
2	Good Link Test Disable
3	IO16 Bug Fix Enable
4	IO CHRDY Bug Fix Enable
5	—
6	Boot PROM Write Enable
7	—
<b>Config C:</b>	
0	Boot PROM Addr and Size
1	Boot PROM Addr and Size
2	Boot PROM Addr and Size
3	Boot PROM Addr and Size
4	RAM Size 8k or 32k
5	4 or 8 Interrupts
6	Core CLK = 20 MHz or BSCLK
7	Allow Access to Configure Regs(1)

**Note 1:** Config C bit 7 "SOFTEN" allows configuration software to read and write config registers A & B. Any changes to the config registers are overwritten by the EEPROM or jumper configuration the next time that the AT/LANTIC is fully reset. If access to the config registers is disabled, it is also not possible to change the EEPROM contents.

To implement a Jumperless (EEPROM) solution, the EECONFIG pin should be tied to +5V. The default configuration must be programmed into the EEPROM during manufacture. See Section 1.3 for details of how to program the EEPROM.

If config register C bit 7 "SOFTEN" is "0":

1. Config registers A and B can be examined by software (to check the configuration).
2. Config registers A and B can be modified by software to temporarily change the adapter settings. The settings are restored to the values in the EEPROM when the AT/LANTIC is next fully reset.
3. The values for config registers A, B and C that are held in the EEPROM can be changed by software. This can include setting config register C bit 7 to "1", which will prevent further access to config registers. If this is done, the settings can only be changed again if AT/LANTIC is reset with EECONFIG pulled low (i.e., jumpered mode).

If config register C bit 7 "SOFTEN" is "1":

1. Config registers A and B cannot be read or written by software.
2. The values for config registers A, B and C that are held in the EEPROM cannot be changed by software.

To implement a Jumpered solution, the EECONFIG pin should be tied to 0V. For each configuration bit you must define the state of the corresponding AT/LANTIC pin at reset. If the bit is fixed by design, you should connect 47k or 10k pull-up for a "1", or leave alone for a "0". If it is user-selectable, you should connect a pull-up via a jumper.

If config register C bit 7 "SOFTEN" is "0":

1. Config registers A and B can be examined by software (to check the configuration).
2. Config registers A and B can be modified by software to temporarily change the adapter settings. The settings are restored according to the jumpers when the AT/LANTIC is next fully reset.

If config register C bit 7 "SOFTEN" is "1":

1. Config registers A and B cannot be read or written by software.

#### OTHER CONFIGURATION OPTIONS

There are two other methods of configuring the AT/LANTIC. If using these methods, the interface will not properly emulate NE2000 adapter cards, and will not work with standard drivers.

**Partially Jumpered Solution:** This is a jumpered solution where only some of the options are jumpered, e.g., I/O address and boot PROM address. Since configuration registers A and B can be rewritten by software, a program could be used to complete the configuration process as part of the software boot sequence.

**No EEPROM, partially jumpered solution:** This is the same as the above solution, except that the EEPROM is not fitted. The purpose is to reduce cost. In this case, there is no Ethernet node address available, so a non-standard driver is required, which obtains its Ethernet address from another source.

**1.2 Layout Considerations**

**1.2.1 ISA Bus**

In a multi-layer PCB with good ground plane, there should be little risk of difficulty due to layout. However, if a 2 layer design is envisaged, a lot of care is required for the ground track routing. The AT/LANTIC drives 16 data lines (SD0-15) with fast high-current drivers. Ground routing should be arranged as a grid, so that there is a good return current path between the AT/LANTIC and all the ISA bus 0V connections.

**1.2.2 Twisted Pair (TPI)**

The length of tracks in the TPI circuit should be kept short and straight, and the lengths of the differential signal tracks

should be kept approximately equal. If using a multi-layer board, the power and ground planes should be removed in the area of the TPI to reduce capacitive coupling of noise from the power planes. The area should not have other signals passing through it. It may be desirable to have chassis ground used around the area of the connector to provide a shield for EMI noise radiation.

Tracks between the filter/isolation transformer and the RJ45 "telephone" socket should have an isolation barrier of at least 2mm to any other track or component.

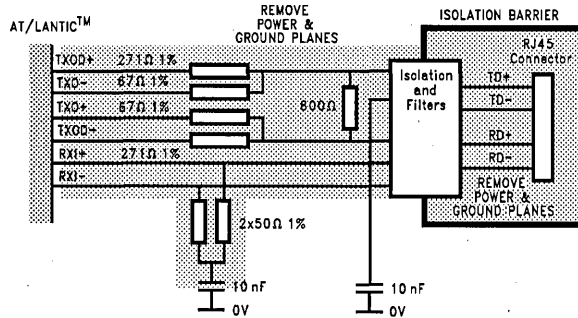


FIGURE 11. TPI Layout

TL/F/11850-11

**1.2.3 Attachment Unit Interface (AUI)**

Tracks should be kept short and straight where possible. The D-type shell (and the metal bracket it is fitted to) should be connected to digital ground via a 10 nF capacitor.

**1.2.4 Thin Ethernet (Coax)**

The DP8392C Coax Transceiver Interface should be placed close to the BNC connector, so that the connection between them is short. There should be no other tracks in this area, and on multi-layer PCBs the power and ground layers should be removed.

The DP8392CV requires an area of copper on the PCB surface to act as a heatsink. This is documented in the DP8392CV data sheet.

All the coax interface components between the isolation transformer and the BNC must be surrounded by an isolation barrier of at least 2mm, which must include the power and ground layers of a multi-layer PCB. The isolation transformer and the DC-DC converter bridge the isolation barrier, plus a resistor, a capacitor and a spark gap. On a multi-layer PCB the isolated power and ground planes should be used for VEE and ISOLATED\_GND.

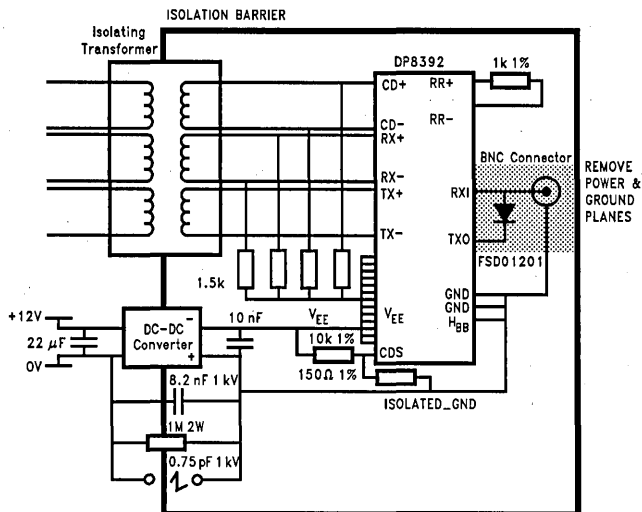
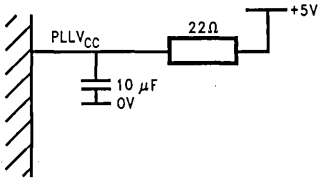


FIGURE 12. Coax Layout

TL/F/11850-12

**1.2.5 PLL**

In order to improve performance of the receiver PLL, the PLL supply pin should be connected via a simple RC filter, located close to the AT/LANTIC PLLV<sub>CC</sub> pin.



TL/F/11850-13

**FIGURE 13. PLL Supply Decoupling**

**1.3 EEPROM Programming**

Initial values must be placed in the EEPROM before it can be used. The following table shows the EEPROM address map (all values in HEX):

Addr	Bits 15-8	Bits 7-0
00	Node Addr 1	Node Addr 0
01	Node Addr 3	Node Addr 2
02	Node Addr 5	Node Addr 4
03	Checksum	05 (8013 Type)
04	00 (Not Used)	00 (Not Used)
05	00 (Not Used)	00 (Not Used)
06	00 (Not Used)	00 (Not Used)
07	57 (ASCII "W")	57 (ASCII "W")
08	42 (ASCII "B")	42 (ASCII "B")
09	00 (Not Used)	00 (Not Used)
0A	00 (Not Used)	00 (Not Used)
0B	00 (Not Used)	00 (Not Used)
0C	00 (Not Used)	00 (Not Used)
0D	00 (Not Used)	00 (Not Used)
0E	Config B	Config A
0F	73H (Note 1)	Config C

**Note 1:** In initial documentation in the data sheet this byte was listed as having a value of FFH. In order to accommodate future expansion of features on the AT/LANTIC, this byte should be programmed with a 73H.

In a jumperless solution, Config A, B and C values must be set according to the default configuration, as discussed in Section 1.1.9.

Config A, B and C are ignored in a jumpered configuration. The Checksum is calculated so that the least significant byte of the sum of the first 8 bytes in the EEPROM is FF (hex).

i.e., ( Node Addr 0  
 + Node Addr 1  
 + Node Addr 2  
 + Node Addr 3  
 + Node Addr 4  
 + Node Addr 5  
 + 05  
 + Checksum ) and FFh = FFh

The Ethernet node address must be unique to each unit produced.

Note that only Config A, B and C bytes can be changed by user software.

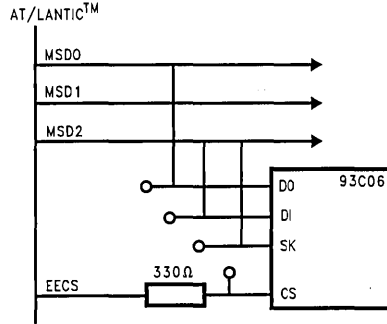
**How to Program the EEPROM**

The simplest way to program the EEPROM is before it is fitted into the PCB.

In some cases, however, it may be preferable to be able to program the EEPROM after board manufacture. This requires either a special programming connector on the PCB, or a "bed of nails" programming jig to access signals on the board. Additionally, a series resistor is required in the EECS signal between AT/LANTIC and the EEPROM.

To program the EEPROM, power must be applied to the PCB and the RESET pin must be forced high on AT/LANTIC. You can then apply the programming waveforms to the EEPROM.

Refer to the 93C06 data sheet for programming details.



TL/F/11850-14

**FIGURE 14. EEPROM In-Situ Programming**

**1.4 Additional End-User Requirements**

The end user may require some of the following items:

**1.4.1 Configuration Software**

If you are offering a jumperless solution, the user will require configuration software. National Semiconductor is able to offer source code for a configuration software package.

Configuration software is unnecessary for jumpered solutions.

**1.4.2 Driver Software**

You may wish to offer drivers with your solution. National Semiconductor offers a selection of drivers for popular network operating systems—contact your representative for the latest list. At the time of printing, National Semiconductor offers the following drivers:

- Novell Network ODI for DOS
- Novell Network ODI for OS/2
- Novell Network ODI for server
- NDIS 2.0
- NDIS 3.0 (available soon)
- SCO UNIX
- PC TCP Packet Driver

**1.4.3 Documentation (Installation Guide)**

We suggest that your documentation includes the following topics:

**Configuration**

If you have a jumpered solution, the user will have no software assistance, and you will have to document how to choose addresses and interrupts without causing conflicts, and how to set the jumpers.

In a jumperless solution, some assistance is given by the configuration program in determining addresses and interrupts currently in use. You will have to document how to run the software, and, for adapter cards, how to cope with installation into systems that require manual installation (for example, it may not be possible to detect I/O space usage if there are no pull-ups on the data bus).

### Bus Compatibility Modes

Some PC's use chip sets that have impossible timing requirements. If this is the case, the driver or configuration software tests will report a problem with the buffer RAM.

In machines with this timing problem, AT/LANTIC offers two "fixes". We recommend trying the "IO16" fix first. If that is not successful, use the "CHRDY" fix instead.

See Section 2.3.6 for a description of how these "fixes" work.

## SOFTWARE INSTALLATION

You may wish to describe how to install the configuration and driver software in a typical application.

## 2.0 REFERENCE INFORMATION

### 2.1 Architecture

#### 2.1.1 NIC Core

The AT/LANTIC controller contains a DP8390 NIC (Network Interface Controller) core. This controller was used as a discrete part on the NE2000 and WD Ethercard Plus adapter cards. The operation of the NIC core is fundamental to the NE2000, WD Plus and consequently the AT/LANTIC.

All packets transmitted and received are sent via a dedicated buffer memory. A "local" DMA controller within the NIC transfers the data between the buffer RAM and the NIC's serializer/deserializer. This approach means that there are no critical performance requirements placed on the host computer (i.e., the PC).

At initialization time, the network software (the driver) tells the NIC to reserve a section of the buffer RAM for receive packet data. Since packets may be received without warning, the receive buffer is usually as large as possible. The NIC uses the receive buffer as a cyclic buffer, and maintains hardware pointers to put data in the correct place and ensure that data not yet read by the host is not overwritten.

When transmitting a packet, the host places the data into the buffer RAM (not in the receive buffer area), and issues a transmit command to the NIC, specifying RAM start address and length.

The host may access the buffer RAM in one of two ways:

**Using the remote DMA channel:** The NIC has another DMA controller for host transfers. This allows the host to transfer a block of data to or from the buffer RAM by writing to or reading from a single data transfer port. The NE2000 adapter uses this technique. The data transfer port is I/O mapped, and this method is often called "I/O MODE".

**Memory mapped:** Alternatively, the buffer memory can be mapped into the host address space. This allows the host to directly access any buffer RAM location. Each access requires arbitration between the NIC and the host. This method is used by the WD Plus adapter, and is commonly referred to as "SHARED MEMORY MODE" because the memory is shared between the NIC and the host.

In the AT/LANTIC, all arbitration and handshaking is handled internally for both I/O and Shared Memory modes.

### 2.1.2 NE2000

The key features of the NE2000 are:

1. The NIC registers and the data transfer port are I/O mapped on the ISA bus.
2. The boot PROM is memory mapped on the ISA bus.
3. There is an Ethernet address PROM mapped onto the NIC buffer RAM bus. The PROM is 32 bytes, but only 16 bytes can be read. Which 16 bytes depends on whether the card is in an 8- or 16-bit slot.

In the AT/LANTIC, the Ethernet address PROM is implemented as registers, which are loaded at reset from the EEPROM.

### 2.1.3 Shared Memory Mode

The key features of the shared memory mode are:

1. NIC core and "shared memory control registers" are I/O mapped on the ISA bus.
2. A PROM containing Ethernet address is also I/O mapped on the ISA bus.
3. Buffer memory is memory mapped on the ISA bus at an address determined by the "shared memory control registers".
4. The boot PROM is memory mapped on the ISA bus.

In the AT/LANTIC, the Ethernet address PROM is implemented as registers, which are loaded at reset from the EEPROM.

It should be noted that although the shared memory mode is hardware compatible with the WD Plus architecture, drivers written by WD/SMC check for a specific IEEE address range before enabling the driver.

## 2.2 Memory and I/O Maps

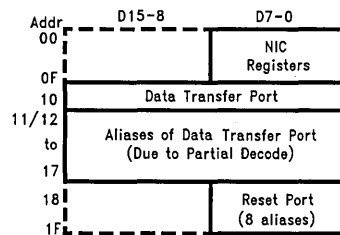
Please refer to Section 5 of the AT/LANTIC data sheet for details of how to use the registers shown here.

### 2.2.1 NE2000 8-Bit and 16-Bit

The ISA I/O map comprises a block of 32 addresses which can be located at one of 7 base addresses (240h, 280h, 2C0h, 300h, 320h, 340h and 360h).

The NIC registers and reset port are BYTE wide.

The data transfer port is the same width as the interface.



TL/F/11850-15

The NIC buffer memory map varies according to the RAM size, and also the interface width as programmed into an NIC register. This register should be set according to whether the interface is 8- or 16-bit.

Note that an 8 bit interface can be either an 8-bit design or a 16-bit adapter in an 8-bit slot.

**8k x 8 RAMs, 16-Bit Interface**

Addr	
0000	PROM
001E	
0020	Aliases of PROM
3FFE	
4000	16 kbytes of Buffer RAM
7FFE	
8000	Alias of 0000 to 7FFE
FFFE	

**8k x 8 RAMs, 8-Bit Interface**

Addr	
0000	PROM
001F	
0020	Aliases of PROM
3FFF	
4000	8 kbytes of Buffer RAM
5FFF	
6000	Alias of Buffer RAM
7FFF	
8000	Alias of 0000 to 7FFF
FFFF	

**32k x 8 RAMs, 16-Bit Interface**

Addr	
0000	PROM
001E	
0020	Aliases of PROM
00FE	
0100	63.75 kbytes of Buffer RAM
FFFE	

**32k x 8 RAMs, 8-Bit Interface**

Addr	
0000	PROM
001F	
0020	Aliases of PROM
00FF	
0100	Alias of Buffer RAM
7FFF	
8000	32 kbytes of Buffer RAM
FFFF	

The PROM is always read-only.

In a 16-bit configuration, the PROM is arranged as follows:

Addr	D15-8	D7-0
00	Invalid	Node Addr 0
02	Invalid	Node Addr 1
04	Invalid	Node Addr 2
06	Invalid	Node Addr 3
08	Invalid	Node Addr 4
0A	Invalid	Node Addr 5
0C		
to	Invalid	00h
1A		
1C	Invalid	42h/57h*
1E	Invalid	42h/57h*

In an 8-bit configuration, the PROM is arranged as follows:

Addr	D7-0
00	Node Addr 0
01	Node Addr 0
02	Node Addr 1
03	Node Addr 1
04	Node Addr 2
05	Node Addr 2
06	Node Addr 3
07	Node Addr 3
08	Node Addr 4
09	Node Addr 4
0	Node Addr 5
0A	Node Addr 5
0C	
to	00h
1B	
1C	42h/57h*
1D	42h/57h*
1E	42h/57h*
1F	42h/57h*

\*The value found at PROM addresses 1C to 1F is determined by the logic level on the DWID pin:

- 42h ("B") if DWID = 0 8-bit (byte) interface
- 57h ("W") if DWID = 1 16-bit (word) interface

The NIC must be programmed for 8- or 16-bit operation before it is possible to read the PROM. It is therefore normal practice for software to set 8-bit mode and read the PROM using 8-bit I/O instructions in order that the actual bus size can be determined from the value at PROM address 1C-1F.

**2.2.2 Shared Memory Mode**

The ISA I/O map comprises a block of 32 addresses which can be located at one of 7 base addresses (240h, 280h, 2C0h, 300h, 320h, 340h and 360h).

All the registers are BYTE wide.

Addr	D7-0	
00	Control 1	
01	AT detect	(Read Only)
02	Unused	
03	Unused	
04	Unused	
05	Control 2	
06	Unused	
07	Unused	
08	Node Addr 0	(Read Only)
09	Node Addr 1	(Read Only)
0A	Node Addr 2	(Read Only)
0B	Node Addr 3	(Read Only)
0C	Node Addr 4	(Read Only)
0D	Node Addr 5	(Read Only)
0E	05h	(Read Only)
0F	Checksum	(Read Only)
10 to 1F	NIC registers	

The NIC buffer memory map varies according to the RAM size, and the bus width which is programmed into an NIC register. This should be set according to whether the interface is 8- or 16-bit. The bus size is detected by the hardware on the DWID pin. This information is available to software by reading the AT Detect register.

Note that an 8-bit interface can be either an 8-bit design or a 16-bit adapter in an 8-bit slot.

**8k x 8 RAMs, 16-Bit Interface**

Addr	
0000	16 kbytes of Buffer RAM
3FFE	
4000	Aliases of Buffer RAM
FFFE	

**8k x 8 RAMs, 8-Bit Interface**

Addr	
0000	8 kbytes of Buffer RAM
1FFE	
2000	Aliases of Buffer RAM
FFFE	

**32k x 8 RAMs, 16-Bit Interface**

Addr	
0000	64 kbytes of Buffer RAM
FFFE	

**32k x 8 RAMs, 8-Bit Interface**

Addr	
0000	32 kbytes of Buffer RAM
7FFE	
8000	Alias of Buffer RAM
FFFE	

**2.3 ISA Bus Description**

This section is a brief tutorial for those unfamiliar with the ISA bus.

For the most part, the ISA bus should be considered asynchronous. There are two ISA bus address spaces: I/O and memory.

**2.3.1 I/O Cycles**

I/O cycles are generated when the CPU performs IN or OUT instructions. An I/O cycle is signalled on the bus by IORD (read) or IOWR (write) strobes being active (low). During IORD or IOWR cycles, the address is given on SA0-9. I/O devices must qualify the IORD or IOWR strobe with a valid address, and the AEN signal, which must be low.

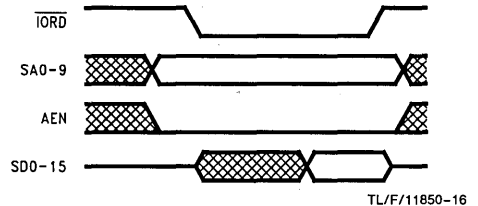


FIGURE 15. I/O Read Cycle

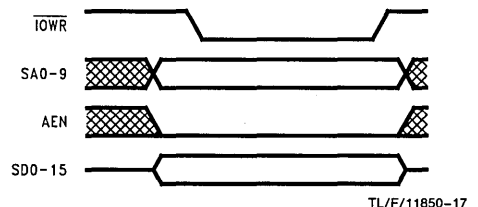


FIGURE 16. I/O Write Cycle

**2.3.2 Memory Cycles**

Memory cycles are generated when the CPU performs MOV instructions. In most PC systems, memory accesses to system RAM are private to the motherboard and do not cause ISA bus activity.

Thus it is usual to only see accesses to devices on the ISA bus.

Similar to I/O cycles, memory cycles have memory read and write strobes. Unlike I/O cycles, however, there are two of each.

For an 8-bit adapter card, memory strobes SMRD and SMWR are used, together with address SA0-19. This allows the memory device to decode any address range in the bottom 1 Mbyte of the CPU address space. SMRD and SMWR strobes are not active for memory accesses above 1 Mbyte ( $\geq 100000h$ ).

For a 16-bit adapter, memory strobes MRD and MWR are used. These are active for memory accesses to any address. The adapter card must therefore additionally test address lines LA17-23.

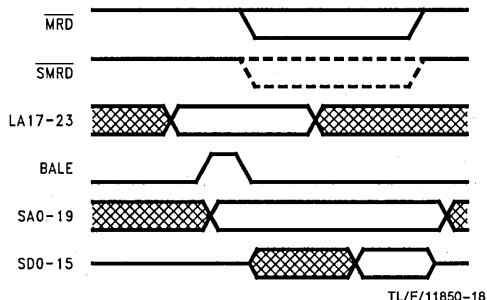


FIGURE 17. Memory Read Cycle

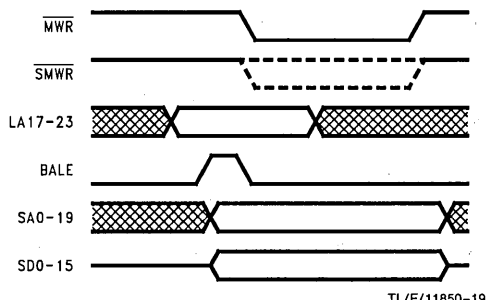


FIGURE 18. Memory Write Cycle

**2.3.3 Cycle Timing**

For all cycles, the system will use a set of default timings. These vary according to whether the access is I/O or memory, and whether the adapter card is 8- or 16-bit.

Adaptor cards can extend any cycle beyond the default timing by driving the CHRDY (Channel Ready) signal to 0V (meaning not ready). The cycle will be extended until CHRDY is released. A pull up on the system board pulls CHRDY high again—adapter cards are not allowed to drive it high.

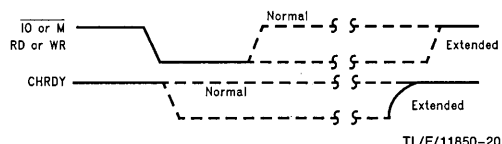


FIGURE 19. Extended ISA Cycle

Some cycles can also be shortened by adapter cards by driving a signal OWS low. This signal is not used by AT/LANTIC.

**2.3.4 8-Bit and 16-Bit Cycles**

For all cycles, the system will assume that an 8-bit device is being accessed. If a 16-bit transfer is requested by the CPU, the system will automatically convert it into two 8-bit cycles. When a 16-bit device is accessed, it informs the system of its presence, so that the data can be transferred in one cycle.

Timing requirements for memory accesses mean that the system needs to know the size of the device being accessed (8- or 16-bits) before the strobe is asserted. At this time, the device does not know whether the address corresponds to I/O or memory space. The ISA bus therefore uses two signals to indicate that a device is 16 bits: IO16 and M16.

IO16 is driven low by a device that detects an address on SA0-9 according to a word I/O port.

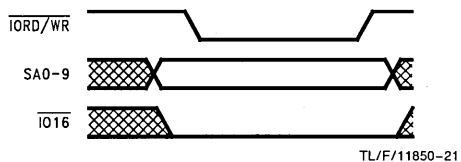


FIGURE 20. IO16 Cycle

M16 is driven low by a device that detects a valid address on LA17-23 for 16-bit memory. LA17-23 are unlatched address lines that are valid earlier than SA0-19. They become invalid before the end of the cycle. M16 is latched by the system. The device is required to latch LA17-23 for use by its address decoder. The BALE signal should be used as the latch enable for the LA address latch.

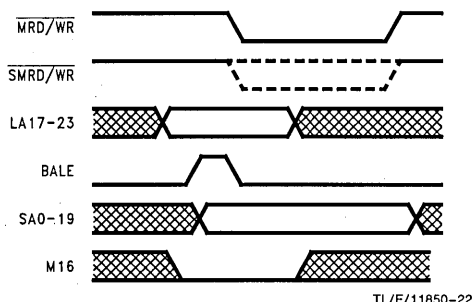


FIGURE 21. M16 Cycle

The size of the system transfer is indicated on signals SA0 and SBHE:

SA0	SBHE	
L	H	Byte transfer, even address
H	L	Byte transfer, odd address
L	L	Word transfer
H	H	Not used

SBHE has the same timing as SA0-19.

**2.3.5 DMA and Refresh Cycles**

DMA cycles are direct transfers between an I/O port and memory. One I/O strobe and one memory strobe are used together for the cycle. The I/O device is selected by one of 7 DACK signals. The memory device is selected by the LA17-23 and SA0-19 address lines. In order that the I/O device corresponding to SA0-9 does not respond, the AEN signal is driven high; I/O devices are only allowed to respond if AEN is low. ISA DMA is not used by AT/LANTIC.

Refresh cycles occur every 15.6  $\mu$ s. They are similar to memory read cycles, except that the REFRESH signal is active. Refresh is not used by AT/LANTIC.

**2.3.6 Bus Timing Compatibility Modes**

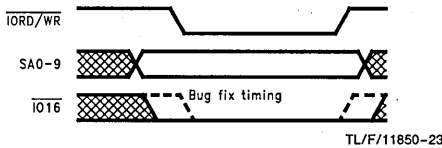
In some PCs using certain chipsets, the timing requirement for an adapter card to drive CHRDY from receiving an active IORD or IOWD is impossible to meet. The consequence is that the ISA bus completes the cycle even though the AT/LANTIC requires more time. AT/LANTIC incorporates logic to detect this condition, and has two ways of overcoming the problem.

If the condition occurs, configuration register B bit 5 "BE" is set to "1". This can be used by software to warn that a fix is required.

The two timing change modes are:

**IO16 Mode**

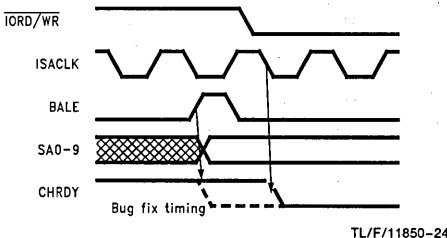
The problem only occurs for 16-bit cycles. IO16 is normally asserted whenever the address SA0-9 is valid. By additionally requiring that IORD or IOWR is asserted before IO16 is driven low, the offending chipsets are fooled into accepting 8-bit timing for CHRDY, while still transferring 16-bits correctly.



**FIGURE 22. IO16 Timing Change Mode**

**CHRDY Mode**

It is possible to drive CHRDY low early, qualified by SA0-9 and AEN without IORD or IOWR. If a valid address is present, and AT/LANTIC requires a longer-than-default cycle, this circuit starts driving CHRDY low as soon as BALE is active. If, at any time, MRD or MWR memory strobes become asserted, then CHRDY is released immediately. Otherwise, CHRDY is held low until the next falling edge of ISACKL after BALE has gone low. In a typical system, this clock edge is immediately followed by IORD or IOWR being asserted. The small "gap" between ISACKL and IORD or IOWR will not normally be long enough for CHRDY to be pulled high by the pull-up resistor.



**FIGURE 23. CHRDY Fix**

**2.4 Boot PROM**

The boot PROM interface uses the AT/LANTIC to decode the ISA address, and to provide data bus drivers onto the ISA bus. Because the PROM data is connected to the NIC memory support data bus, arbitration is required to access the PROM. The boot PROM is not mapped onto the NIC memory map.

If the MSWR signal is connected, a FLASH boot PROM can be used, allowing in-situ programming or updating.

If the PROM is read-only, the configuration register B bit 6 "BPWR" should be set to "0" to prevent bus contention if write cycles are attempted.

The boot PROM appears as an 8-bit device, regardless of the size of the interface. However there is a special case where it will not work properly. When emulating a WD Plus adapter, AT/LANTIC uses 16-bit wide buffer RAM, which it declares as 16-bit to the ISA bus by driving M16 low. The decode logic for M16 uses only LA17-23, i.e., a 128 kbyte region—this is an ISA bus limitation. If the boot PROM and the buffer memory are placed within the same 128 kbyte region, the boot PROM will appear to the ISA bus as a 16-bit device. Since the PROM can only produce 8-bits of data, it will fail.

Boot PROMs can be written to overcome this problem by first copying the PROM contents to system memory, and then executing from the copy instead. The PROM therefore need not be accessed at the same time as the shared RAM is enabled.

**2.5 Boot PROM and RAM Timing Calculations**

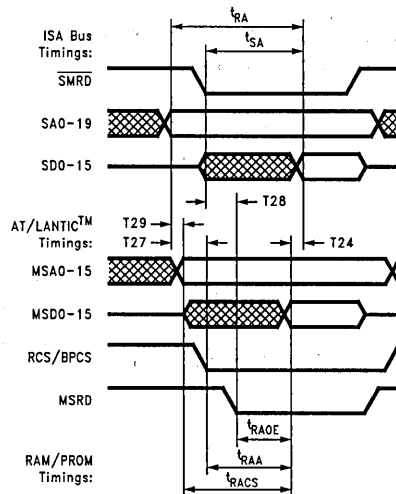
Boot PROM timings are derived from the ISA bus timings minus AT/LANTIC propagation delays.

RAM timings are the worst case of two situations: ISA bus accesses, similar to the Boot PROM, and AT/LANTIC DMA channel accesses.

For 8-bit cards, 8-bit ISA timings should be used for both RAM and boot PROM.

For 16-bit cards, 16-bit ISA timings should be used for the buffer RAM, and 8-bit timings should be used for the boot PROM.

**2.5.1 Boot PROM/RAM ISA Read Timing**



**FIGURE 24. Boot PROM/RAM ISA Read**



Access time from OE:

$$t_{RAOE} = t_{RA} \text{ min} - T28\text{max} - T24\text{max}$$

Access time from address:

$$t_{RAA} = t_{SA} \text{ min} - T29\text{max} - T24\text{max}$$

Access time from CS:

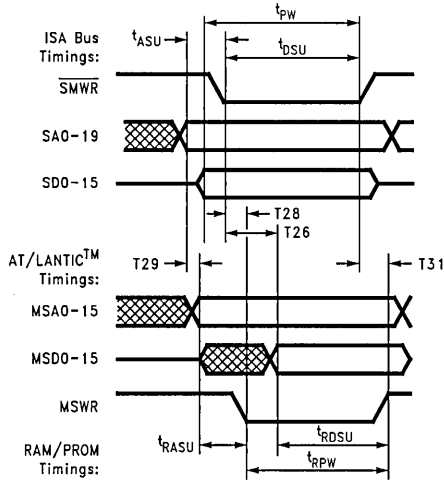
$$t_{RACS} = t_{SA} \text{ min} - T27\text{max} - T24\text{max}$$

For a typical 8 MHz ISA bus:

$$t_{RA} \text{ min} = 480 \text{ ns (8 bit) or } 160 \text{ ns (16 bit),}$$

$$t_{SA} \text{ min} = 570 \text{ ns (8 bit) or } 200 \text{ ns (16 bit).}$$

**2.5.2 Boot PROM/RAM ISA Write Timing**



TL/F/11850-26

**FIGURE 25. Boot PROM/RAM ISA Write**

Address setup time:

$$t_{RASU} = t_{ASU} \text{ min} + t28\text{min} - T29\text{max}$$

Data setup time:

$$t_{RDSU} = t_{DSU} \text{ min} + t31 \text{ min} - T26\text{max}$$

Write pulse width:

$$t_{RPW} = t_{PW} \text{ min} + t31 \text{ min} - T28\text{max}$$

For a typical 8 MHz ISA bus:

$$t_{RASU} \text{ min} = 90 \text{ ns (8 bit) or } 28 \text{ ns (16 bit),}$$

$$t_{RPW} \text{ min} = 530 \text{ ns (8 bit) or } 154 \text{ ns (16 bit),}$$

$$t_{RDSU} \text{ min} = 470 \text{ ns (8 bit) or } 105 \text{ ns (16 bit).}$$

**2.5.3 RAM DMA Timing**

These timings are given in the AT/LANTIC data sheet Section 8, under "I/O port or FIFO transfers".

$$t_{RAOE} = T3 - T7,$$

$$t_{RAA} = T2 + T3 - T7,$$

$$t_{RACS} = T4 - T7,$$

$$t_{RASU} = T2,$$

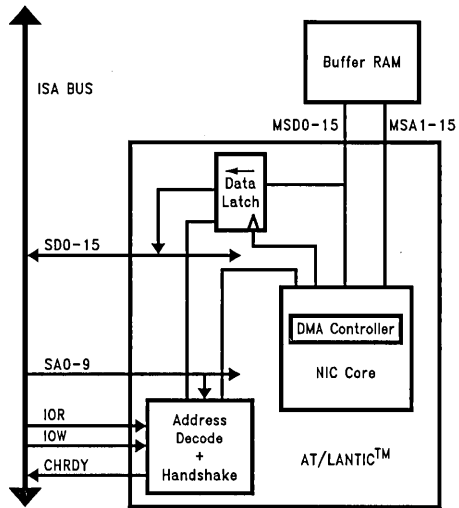
$$t_{RDSU} = T9,$$

$$t_{RPW} = T3$$

**2.6 Fast Read Feature**

This is a feature designed into AT/LANTIC to improve the performance of the remote DMA channel in NE2000 mode. The actual performance measured will depend on the platform that AT/LANTIC is used with, and the test configuration used.

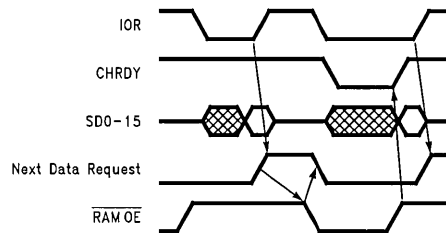
The architecture of the remote DMA channel is as shown:



TL/F/11850-27

**FIGURE 26. Remote DMA Controller**

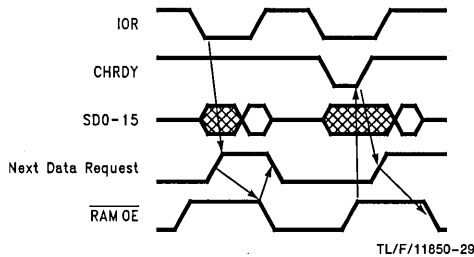
When the remote DMA read is started, by I/O writes to the NIC remote DMA controller, the buffer RAM is accessed at the first address and the data is latched in the data transfer latch. The PC then reads this data over the ISA bus. At the end of the IORD strobe, the DMA controller fetches data from the next RAM address and latches it into the latch. Hardware handshake logic prevents the PC from completing a read cycle until the data is ready.



TL/F/11850-28

**FIGURE 27. Remote DMA Read Timing**

The request for more data from the DMA controller is not made until the end of the IORD strobe. When fast read operation is enabled, the request for more data is made as soon as CHRDY is released, before the end of the IORD. For cycles that do not require CHRDY, i.e. the data is already in the latch, more data is requested at the start of the IORD strobe.



**FIGURE 28. Fast Read Remote DMA Timing**

There is a danger, in very slow machines, that the new data will be latched into the data transfer latch before the end of the current IORD cycle. For this reason, Fast Read is not recommended for 8-bit systems.

The NIC core is specified to take a minimum of 11 clocks between data being requested and the latch being updated. A typical system with an NIC core clock of 20 MHz will therefore take at least 550 ns. Fast Read mode is safe in any system where the IORD strobe width, or the CHRDY to IORD high delay does not exceed 550ns.

## 2.7 Reset Operation

AT/LANTIC has different degrees of RESET according to the duration of the RESET pulse. The pulse width is measured by counting X1 clocks (20 MHz), so the timing period can only begin after the oscillator has started.

In order to prevent noise problems, a RESET pulse of less than 350 ns will be ignored. The RESET pulse must be at least 400 ns to be guaranteed to be recognized.

A RESET pulse of at least 400 ns will reset the internal logic, including the 8390 NIC core, and will tristate all I/O pins. A 60k pull down will be enabled for each configuration pin MSD0-15 and MSA1-8.

If the RESET pulse is more than 400 ns long, configuration data and Ethernet node address will be loaded. The load sequence begins when RESET goes low, and can take up to 320  $\mu$ s. During this time, all ISA bus cycles are ignored.

# Layout Guidelines for an Ethernet Adapter Using the DP83907 AT/LANTIC™ II

National Semiconductor  
Application Note 973



AN-973

## OVERVIEW

This application outlines board layout and design considerations when using the DP83907.

## Board Design and Layout

In a multi-layer PCB with good ground and power planes, there should be little risk of noise problems due to layout. When designing a two-layer board special care is needed. The DP83907 drives 16 data lines (SD0-15) with fast high-current drivers that will cause ground bounce if proper care is not taken in laying out the VCC and GND lines. Ground routing should be arranged in a grid to ensure good return current paths between the DP83907 and all the ISA bus ground connections. Analog and Digital VCC and GND lines should not overlap to prevent crosstalk noise.

### 1.0 Crystal and Oscillator Design

Care should be taken when connecting a crystal. Stray capacitance (e.g., from PC board traces and plated-through holes around X1 and X2 pins) can shift the crystal's frequency out of range, causing the transmitted frequency to exceed the 0.01% tolerance specified by the IEEE. The crystal or oscillator layout should locate all components close to the X1 and X2 pins and should use short traces to avoid excess capacitance and inductance. When using a standard size crystal the crystal should be laid flat with power planes voided in that area. When using a low-profile crystal this is not necessary.

### 1.1 Decoupling for the DP83907

The DP83907 is composed of multiple functional blocks. The analog and digital functional blocks should be run on separate power rails.

Since the power supply pins on the DP83907 are all connected to the same 5V/0V supply in the ISA connector, noise from other slots in the system will be coupled to the power supply pins on the DP83907. This problem is particularly serious in two-layer boards having no separate power or ground planes. To minimize the impact, the DP83907 should be run on separate VCC and ground traces whenever possible. These traces should join near the PC ISA bus connector and be properly filtered.

The first area which requires attention is the switching on the DP83907's System Data (SD) bus. This bus provides high current drive, TRI-STATE®, fast rise and fall times and is a major noise contributor. The VCC pins which provide power to SD bus drivers and the GND pins which are the return path should be carefully decoupled to reduce the

noise. In order to reduce noise the power supply pins and the ground pins on the DP83907 should be decoupled by 0.1  $\mu$ F and 1000 pF capacitors, with the capacitors connected as close as possible to the chip and the plane.

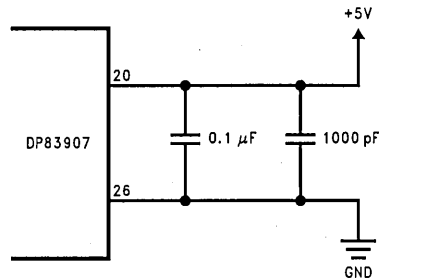


FIGURE 1. Power and Ground Decoupling

This should be done on all the VCC and GND pairs which are pins 20-23, 26-27, 48-49, 35-32, 70-75, 103-100, 109-110. Exceptions are analog VCC and GND pins 40 and 41 which have different decoupling requirements and are discussed in Section 1.2.

The rest of the GND pins except for pin 41 (AGND) should be directly connected to the ground plane.

The second area where care must be taken is in isolation of "quiet" output and input signals from "noisy" output signals. In this regard, the SD outputs should be a minimum of a double space away from ANY other signal trace, preferably with a ground island in between. Similarly, the IRQ traces should be kept isolated from other signals.

Additionally, care must be taken to prevent fast control signals (IOCHRDY, IO16) causing crosstalk on sensitive inputs. To this end, IO16 must be separated by a double space from input signals (SA0, SA1 ...). In the case of IOCHRDY, the ground pin supplying IOCHRDY (pin 14) must be connected through a ferrite bead (26Z @ 100 MHz) and IOCHRDY must be separated from AEN by at LEAST a double space, preferably with an isolation ground island placed in between. This island should connect to 0V only at the edge connector.

These layout considerations have been taken into account in the latest DP83907EB-AT board design. Manufacturers must follow these recommendations in their design to avoid possible noise related problems.

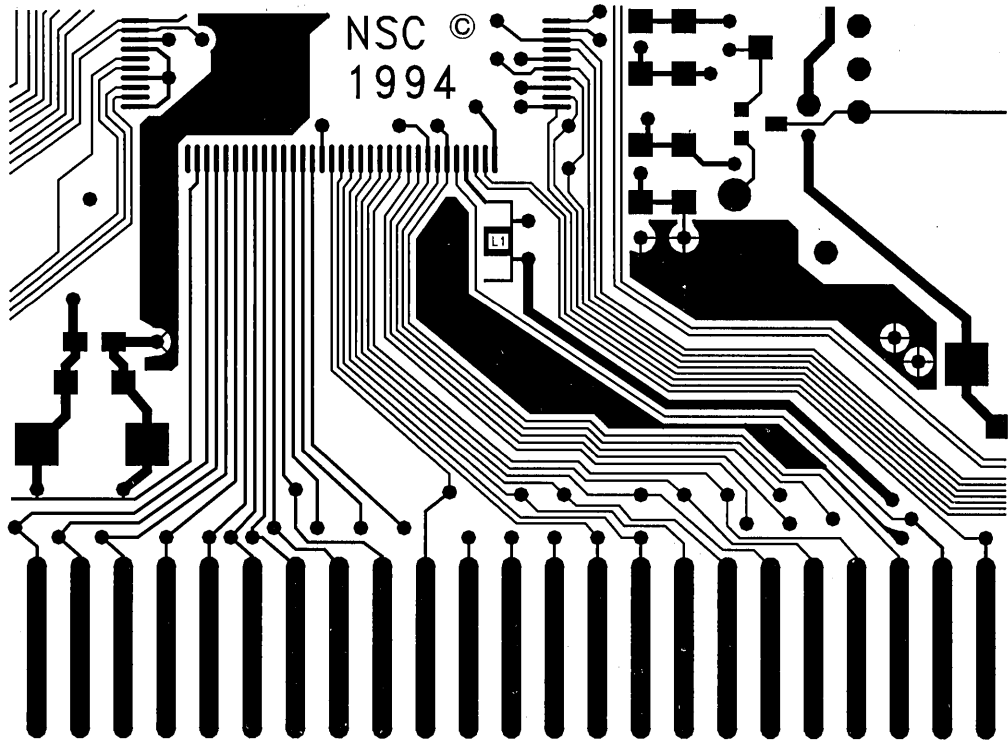
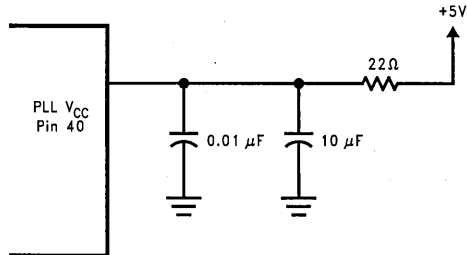


FIGURE 2. Signal Isolation Detail View

TL/F/12334-4

### 1.2 PLL Power Supply Noise

In order to improve the performance of the PLL, a single pole filter should be used on the PLL power supply pin. This is shown on *Figure 3*.

FIGURE 3. PLL  $V_{CC}$  Noise Filter

TL/F/12334-2

### 1.3 PC ISA Bus Interface

Decoupling of noise from the ISA Bus power supply is achieved using 0.1  $\mu\text{F}$ , 100 pF and 22  $\mu\text{F}$  capacitors (used to filter out lower frequency noise in the order of a few kHz). All these three capacitors are placed near the ISA bus connector. This is done for both power supplies of the ISA bus.

*Care should also be taken to place the DP83907 as close to the edge connector as possible, with the aim of keeping the ground tracks to a minimum length.*

### 1.4 Attachment Unit Interface

The AUI signals are differential signals. Each pair of these differential signals should be routed in adjacent channels. The traces should go as straight as possible and avoid forming loops to minimize the potential of magnetic coupling. It is not necessary to void the power plane under them.

### 1.5 Twisted Pair Interface

The transmit signals should be of the same length and as close as possible. These signals are TXOD+, TXOD-, TXO+ and TXO-.

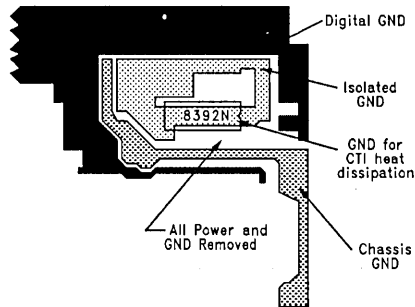
The receive signals should also be of the same length and as close as possible.

The transmit and receive signals should be kept away from each other.

The resistors should be directly tied between the RTX and REQ pins to TPVCC or TPGND.

### 1.6 Coaxial Transceiver Interface (CTI)

The Coax Transceiver Interface (DP8392CN) should be placed very close to the BNC connector. This will allow the TXO/RXI trace to be short and straight. The area under these traces should be voided of all other signal and power planes (as shown in *Figure 4*) in order to meet the IEEE 802.3 input capacitance requirements. The BNC connector on the evaluation board is a standard part, however, "quiet" connectors are readily available for better noise figures and FCC qualification. The CTI requires an area of copper for heat dissipation. This is also shown in *Figure 4* and is documented in the DP8392C data sheet. The ground shield of the BNC connector and coax cable are resistively and capacitively decoupled to chassis ground. There is a chassis ground strip on each layer of the board. This is connected to the chassis of the PC through the 15-pin AUI connector's metal body. Chassis ground is then capacitively decoupled to digital ground through capacitors. The chassis ground trace along the front of the board forms a "shield" so that noise is not emitted into or received from the environment.



TL/F/12334-3

FIGURE 4. DP83907EB-AT CTI Layout Diagram

### 1.7 FCC Considerations

For applications requiring FCC class B certification it is strongly recommended that the existing BNC connector be replaced with a *filtered* BNC connector and capacitors C45 and C54 removed.

A filtered BNC connector similar to p/n 413515 from AMP incorporated or p/n 456-117 from Amphenol Corp. (or equivalent) will suffice.

# Design Guide for an Ethernet Adapter Using the DP83907 AT/LANTIC™ II

National Semiconductor  
Application Note 974



## TABLE OF CONTENTS

### 1.0 INTRODUCTION

### 2.0 EVALUATION BOARD FEATURES

### 3.0 HARDWARE DESIGN CONSIDERATIONS

- 3.1 Crystal and Oscillator
- 3.2 ISA Bus Interface
- 3.3 Memory Support Interface
  - 3.3.1 Buffer SRAM
  - 3.3.2 EEPROM
- 3.4 Boot ROM
- 3.5 Interrupt Scheme & Status LEDs
- 3.6 Configuration
  - 3.6.1 Default Configuration
  - 3.6.2 Configuration Modification
  - 3.6.3 IO16 Mode
  - 3.6.4 Jumperless Operation Support

### 1.0 INTRODUCTION

The DP83907 AT/LANTIC II Evaluation Board provides system designers a simple method of interfacing PC ISA (Industry Standard Architecture) bus based system to Ethernet. This Evaluation board provides complete 16-bit 10 Base-T, 10Base2 and 10Base5 Ethernet solutions in a half-size jumperless ISA adapter card. It is IEEE 802.3 compliant and NE2000/NE2000Plus I/O mode compatible.

Built into the DP83907 is an Ethernet Media Access Control unit, a Manchester Encoder/Decoder, Twisted Pair Transceiver, an AUI, ISA bus, EEPROM and SRAM interfaces. It uses the EEPROM to store the board's configuration and IEEE node address and two 8kx8 SRAMs to buffer transmit and receive packets. The 10Base2 network interface is implemented with the addition of a 1:1 pulse transformer, a DC-DC Converter and a DP8392 Coaxial Transceiver Interface (CTI). Refer to the block diagram in *Figure 1*.

### 3.7 Twisted Pair (TPI)

- 3.7.1 ON-CHIP Filters.
- 3.7.2 RTX Resistor
- 3.7.3 Non Cat3 Cable

### 3.8 Attachment Unit Interface (AUI)

### 3.9 UTP/STP Function

### 3.10 Auto-Switch Function

### 4.0 SOFTWARE SUPPORT

- 4.1 Configuration & Diagnostics Software
- 4.2 EEPROM in-situ programming
- 4.3 Driver Software

### 5.0 REFERENCES

#### About This Guide

This guide describes the DP83907 AT/LANTIC™ II Evaluation Board and the underlying hardware design considerations

### 2.0 EVALUATION BOARD FEATURES

- Designed with the DP83907.
- Jumper-less design.
- Half-size PC AT adapter card.
- NE2000 / NE2000Plus I/O mode compatible.
- 10Base2 connectivity.
- 16 kByte SRAM packet buffer.
- Serial EEPROM stores the board's configuration and IEEE node address.
- Serial EEPROM can be programmed in-situ.
- Boot ROM socket to allow disk-less boot from NetWare, Lan Manager and other network operating systems.
- 8 interrupts.
- Status LEDs for transmit, receive and collision.

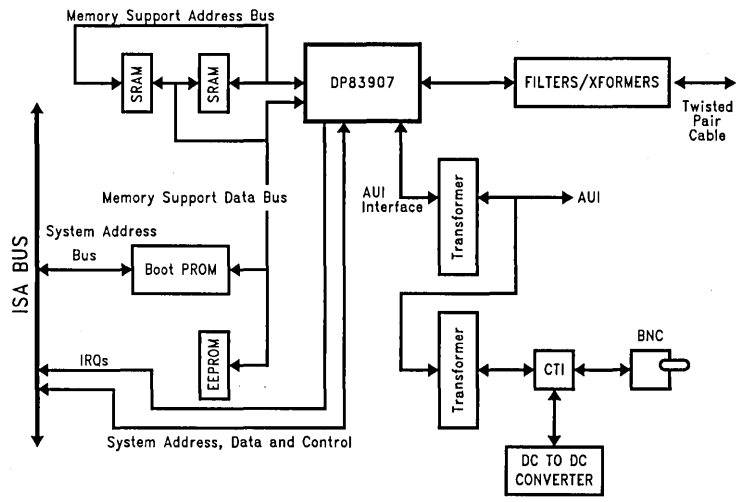


FIGURE 1. DP83907EB Block Diagram

TL/F/12335-1

**3.0 HARDWARE DESIGN CONSIDERATIONS**

This section describes the signal interfaces with the DP83907 in designing this Evaluation Board.

**3.1 Crystal and Oscillator**

The DP83907 has been designed to operate with either a crystal or an oscillator module. The Evaluation Board comes assembled with the crystal option. The crystal should conform to the following specifications.

- AT cut parallel resonance crystal
- Series resistance  $\leq 40\ \Omega$
- Specified load capacitance  $\leq 20\ \text{pF}$
- Accuracy: 50 ppm
- Typical load:  $50\ \mu\text{W} - 75\ \mu\text{W}$

Note, the X2 pin is not guaranteed to provide a TTL compatible logic output and should not be used to drive external logic.

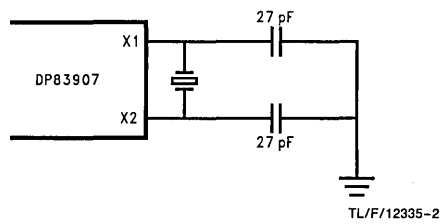


FIGURE 2. Crystal Circuit

If an oscillator module is used, its output should be connected directly to X1, while X2 is left unconnected. The load capacitors should not be used. The oscillator clock should be :

- TTL or CMOS output with a 0.01% freq. tolerance
- 40%–60% duty cycle

**3.2 ISA Bus Interface**

The DP83907EB can be used in 8- or 16-bit ISA slots. 8- or 16-bit mode is determined by MSA9 at reset. For an adapter card this pin can be used to automatically detect if the card has been plugged into an 8- or 16-bit slot by connecting MSA9 via a 10 k $\Omega$  pull-up resistor to V<sub>CC</sub>. When connected to a 16-bit slot MSA9 will be pulled high enabling 16-bit mode. When plugged into an 8-bit slot, MSA9 will be floating enabling 8-bit mode.

For applications requiring 8-bit only interface, MSA9, SD8-15, /IO16 should be left open.

**3.3 Memory Support Interface**

The Memory Support Interface is used by the DP83907's local DMA and auto-configuration. This includes the SRAMs and the EEPROM. Figure 1 shows how the SRAMs and EEPROM are connected to the Memory Support Bus.

On power up, MSA10 must be pulled high to load the EEPROM contents to its internal registers and make them accessible by software. The two 8kx8 SRAMs are used as receive and transmit packet buffers during network data exchanges.

**3.3.1 Buffer SRAM**

The two SRAMs provide 8k words (16-bit) of memory for the DP83907 to buffer received and transmit packets. For an 8-bit-only interface, only one SRAM is needed with the data bus connected to MSD0-7.

For a standard ISA bus configuration, 100 ns SRAMs or faster should be used. Details of how to calculate SRAM speeds can be found in Ref. [3].

**3.3.2 EEPROM**

The DP83907 uses its EECS and MSD0-2 pins for accessing the serial EEPROM (NM93C06). MSD0-2 serve as DO, DI, and SK respectively.

The EEPROM is normally read automatically upon power-on reset. One can modify its content by software thereafter.

There is a possibility that a randomly programmed EEPROM may cause the host system to hang. If this happens, Jumper J2 can be installed (MSA10 pin pulled to GND). This will disable the EEPROM from loading and the DP83907 will initialize to the default settings. The EEPROM can then be correctly programmed by either of the two methods described in section 3.6.2.

**3.4 Boot ROM**

A boot ROM allows the PC to load the operating system from a server on the network without needing a disk drive on the PC (Diskless Workstation). Boot ROMs are available from network operating system vendors or from third parties.

The DP83907EB is supplied with an empty boot ROM socket. Boot ROM sizes of 8kB, 32kB and 64kB are supported. It can be located at particular addresses between C0000h and DFFFFh by programming configuration register C bits 0-3 "BPS0-3". For addresses refer to Reference [1].

The boot ROM address lines must be connected directly to the ISA Bus.

The boot ROM data lines are connected to the DP83907's MSD bus. When the system reads within the selected memory area, the DP83907 reads the data in through MSD0-7 and drives it onto the system data bus. The DP83907 supplies the chip select to the device. See *Figure 3*.

For a standard ISA bus configuration a 250ns ROM or faster should be used. Details of how to calculate boot ROM speeds can be found in Reference [3].

**3.5 Interrupt and Status LEDs**

There are eight interrupt request pins on the DP83907, namely IRQ3-5, IRQ9-12 and IRQ15. The operation of these outputs are determined by configuration register A. The DP83907 has only one Interrupt mode. Configuration Register A controls which one of the 8 interrupt lines will be driven, the others are TRI-STATED. The interrupt outputs should be connected to the following ISA interrupt lines, in the order given, to maintain NE2000 Architecture compatibility: 3, 4, 5, 9, 10, 11, 12 and 15.

**Table I. Direct Drive Interrupt Assignment**

DP83907	ISA Bus
IRQ3	IRQ3
IRQ4	IRQ4
IRQ5	IRQ5
IRQ9	IRQ9
IRQ10	IRQ10
IRQ11	IRQ11
IRQ12	IRQ12
IRQ15	IRQ15

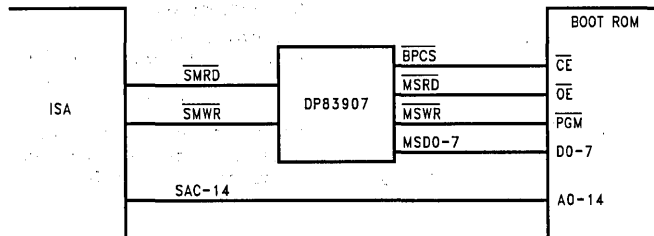
The configuration register A is used for interrupt selection. Bits 3, 4, 5 are set in the following manner for different interrupt lines:

**Table II. Output Interrupt Assignment**

DP83907			ISA Bus
5	4	3	IRQs
INT2	INT1	INT0	IRQs
0	0	0	IRQ 3
0	0	1	IRQ 4
0	1	0	IRQ 5
0	1	1	IRQ 9
1	0	0	IRQ 10
1	0	1	IRQ 11
1	1	0	IRQ 12
1	1	1	IRQ 15

**Status LEDs**

These pins are open-drain, active low outputs and serve as ACT\_LED, GDLNK\_LED and COL\_LED respectively. The maximum sinking current of these outputs are 24 mA. The current limiting resistors should be chosen so that this requirement is not violated. The LED circuit of the DP83907EB is shown in *Figure 5*.



**FIGURE 3. Boot ROM Interface**

TL/F/12335-3



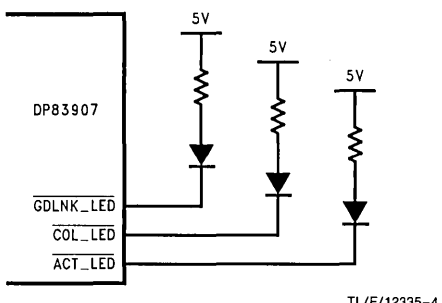


FIGURE 5. LED Circuit

TL/F/12335-4

**3.6 Configuration**

The DP83907 has three configuration registers A, B and C; one signature register and a PROM store register array. All these registers must be properly initialized before the DP83907 can be operational. Table 3-5 may assist you in this task. Note that a full description is given in Reference [1].

**Table III. Configuration Registers**

Config A Register

Bit	Symbol	Use
0-2	IOAD0-2	Select IO address
3-5	INT0-2	Interrupt
6	FRd/Wr	Fast read/write
7	RES	Reserved (must be zero)

Config B Register

Bit	Symbol	Use
0-1	PHYS0- PHYS1	Physical Layer Interface
2	GDLINK	GOOD LINK
3	IO16CON	IO16 bug fix enable
4	RES	Reserved (must be zero)
5	BE	Bus Error Status
6	BPWR	BOOT PROM Write
7	EELOAD	Enable Loading EEPROM

Config C Register

Bit	Symbol	Use
0-3	BPS0-3	Boot ROM addr and size
4	RES	Reserved (must be zero)
5	RES	Reserved (must be one)
6	RES	Reserved (must be zero)
7	SOFEN	Config Reg Access Disable

**Table IV. Signature Register**

Bit	Symbol	Use
0-3	REV0-3	DP83907 Identity (1000)
4	EEPR	Enable EEPROM In-situ Program
5-7	SIG5-7	Board Revision (Note 1)

**Note 1.** This reflects the trapping state of MSA11-13, which can be fixed by manufacturers for board revision identification.

**3.6.1 Default Configuration**

During power-on reset, the DP83907 loads configuration registers A, B and C and bits 5-7 of the signature register from the memory support bus. Subsequently, the DP83907 reads the first 7 words from the EEPROM and maps them into its PROM store. If in 16-bit mode (MSA9 sampled high on power-on reset), it also reads the next word in the EEPROM and appends this. If in 8-bit mode (MSA9 floating), it skips a word, then reads and appends the next word. Mapping from the EEPROM onto the PROM store is shown in the DP83907 data sheet. The EEPROM should be programmed according to Table 5 for NE2000/P<sub>lus</sub> compatibility.

If the MSA10 pin is sensed high, the last two words of the EEPROM are also loaded. Configuration registers A, B and C are re-initialized with the EEPROM content. Hence, a jumper-less solution can be achieved. If the last byte read from EEPROM is 73H, programming in-situ is not allowed. Section 3.6.2 will elaborate more on this.

**Table V. EEPROM CONTENTS (NM93C46/NM93C06)**

	D15	D0
0FH	73H	Config Reg. C
3FH (Note 1)		
0EH	Config Reg. B	Config Reg. A
3EH (Note 1)		
09H-0DH	Not used	Not used
08H	42H	42H
07H	57H	57H
06H	Not used	SUM (Note 2)
05H	Driver ID (Note 3)	0DH
04H	H/W Feature (Note 4)	00H
03H	00H	00H
02H	E'net Addr 5	E'net Addr 4
01H	E'net Addr 3	E'net Addr 2
00H	E'net Addr 1	E'net Addr 0

**Note 1.** If an NM93C46 EEPROM is used then addresses 0FH and 0EH need to be changed to 3FH and 3EH respectively.

**Note 2.** This field contains the sum of the first 6 words (00H-05H).

**Note 3.** Novell Driver ID is 01H.

**Note 4.** This field contains the hardware features of the physical media. The bit 0, 1, 2 should be set respectively for the twisted-pair, thin Ethernet and thick Ethernet.

**3.6.2 Configuration Modification**

For an implementation using jumpers, one can change the jumper setting on the DP83907's memory support bus. This will effectively modify the DP83907's configuration upon power up.

For a jumper-less solution one can program the DP83907's configuration registers A, B and C. Special procedures must be followed as listed in Table 6.

**Table VI. Config Registers Access**

Configuration Registers	Read/Write Requirements	
	Read	Write
A	Page 0 of NIC's registers selected	Preceded by a read on Config A
B	Page 0 of NIC's registers selected	Preceded by a read on Config B
C	Page 0 of NIC's registers selected by 3rd consecutive read of Config A	Preceded by 3rd consecutive read of Config A

To make it a permanent change, one must write the new configuration into the EEPROM. It can be done by either of the two methods described below.

**(1) Indirect Write to EEPROM**

Three bytes of the EEPROM, storing configuration registers A, B and C values, can be modified by a special pseudo code. EELOAD acts as load enable and must be set to one. The DP83907 serializes the data and writes them into the EEPROM. The EELOAD bit (Bit 7 of configuration register B) will be reset by the DP83907 when it completes the load EEPROM operation. It has full control on all the interface signals EECS, DI and SK. Note that this algorithm does not change the configuration registers directly, i.e., the new state only appears on the next power up.

**EEPROM-LOAD ( )**

```

{
    Disable Interrupts( );
    Value = Read(Config_B);
    Value = Value AND GDLINK
    Value = Value OR EELOAD
    Write(Config_B, Value)
    Read(Config_B)
    Write(Config_B, Config_for_A)
    Write(Config_B, Config_for_B)
    Write(Config_B, Config_for_C)
    While(Value AND EELOAD)
    {
        Value = Read(Config_B)
        Wait( );
    }
    Enable Interrupt( )
}

```

In the pseudo code above, interrupt is disabled, GDLINK bit preserved, and the EELOAD bit of configuration B serves as load enable.

**(2) In-situ EEPROM programming**

If the uppermost byte of the EEPROM does not contain 73H, the EEPR bit of the signature register will be zero, and the entire content of the EEPROM can be modified.

One must first set the EELOAD bit (load enable), and subsequent writes to the data transfer port will have SD1-3 driven onto the EECS, SK and DI pins, allowing users to have full control on writing to the EEPROM.

Similar to the previous case, one must apply a power-on reset for the DP83907 to load the new configuration data from the EEPROM.

With in-situ programmability, manufacturers do not have to pre-program the EEPROM before board assembly. It also provides much more flexibility because the entire EEPROM content can be modified at will.

By programming 73H to the uppermost byte, the EEPROM is protected from further changes, except for configuration register A, B and C, which can still be changed with an indirect write to the EEPROM.

**3.6.3 IO16 mode**

In some PCs using certain chip sets, the timing requirements in 16-bit I/O cycle cannot be achieved by the DP83907. As a consequence the host system does not recognize the CHRDY signal and does not insert wait states. The system executes a standard 16-bit cycle and deasserts  $\overline{IORD}$  or  $\overline{IOWR}$  even if the DP83907 is not ready. The DP83907 incorporates logic to detect this condition, Configuration register B bit 5 "BE" is set to "1" if the condition is detected. This can be used by software to warn the user that a fix is required.

By setting bit 3 of configuration register B, the DP83907 enters IO16 mode. It generates  $\overline{IO16}$  after  $\overline{IORD}$  or  $\overline{IOWR}$ . The offending chip sets are fooled into accepting 8-bit timing for CHRDY, while still transferring 16 bits correctly.

**3.6.4 Jumperless Operation Support**

One of the biggest problems in installing new adapters in a PC is not knowing the available resources within that machine. The DP83907 software configuration overcomes that problem. The conflicts possible in the I/O base selection can be overcome by a special mode for software configuration of the I/O base address. By using this mode, and by using the configuration storage capability of the EEPROM, a fully software configurable design on the ISA bus can be realized without address conflict problems.

This mode is invoked by having the DP83907 default to jumper-less software configuration option in the I/O base selection. This mode enables configuration register A to be mapped to address location 278H which is defined to be a printer port's data register. If software writes to this location four consecutive times, on the fourth write the DP83907 will load the data written into the I/O address bits of Configuration Register A. This data should set the I/O base address to a known conflict-free value. The DP83907 can now be configured and operated at the desired base I/O address. If desired, the configuration software could change the EEPROM content to the new values eliminating the need to reconfigure upon each power up. Alternately the software could leave the EEPROM alone and execute the configuration using the printer port's data register upon each power up. This configuration scheme will only work once after each power-up. Therefore the user can not enable the DP83907 from reserved mode, change it back into reserved mode and enable it again. A power-on reset must occur between the first time it is enabled from the reserved mode and the second.

### 3.7 Twisted Pair Interface (TPI)

The Twisted Pair interface is simple, it requires components external to the DP83907 which are pre-emphasis resistors, some capacitors and a transformer/filter module.

#### 3.7.1 ON-CHIP Filters

The on-chip filters are enabled via an external pull-up resistors on MSD12 at configuration. Only an isolation transformer and a few impedance matching resistors are needed for the transmit and receive twisted pair interface.

#### 3.7.2 RTX Resistor

In on-chip filter mode, the value of TPI VOD when measured driving a  $100\Omega$  load may not meet the IEEE-802.3 specification over temperature and process without the use of a resistor of RTX. This does not adversely affect system performance.

An R9 resistor value of  $82.5\text{ k}\Omega$  ( $\pm 1\%$ ) to  $V_{CC}$  is recommended.

- Increasing the value of this resistor will lower the value of VOD.
- Decreasing the value of this resistor will increase the value of VOD.

#### 3.7.3 Non Cat3 Cable

When using non-Cat3 cable such as type 9, 2 pair, 26AWG cable, the following modification is suggested to ensure optimum operating performance.

- Install a  $2200\text{ pF}$  ( $\pm 10\%$ ) capacitor across each  $10.5\Omega$  summing resistor, i.e., R11 and R13.
- Remove any R9 resistor.
- Install a  $24.3\text{ k}\Omega$  ( $\pm 1\%$ ) resistor from RTX to GND, i.e., R10.

### 3.8 Attachment Unit Interface (AUI)

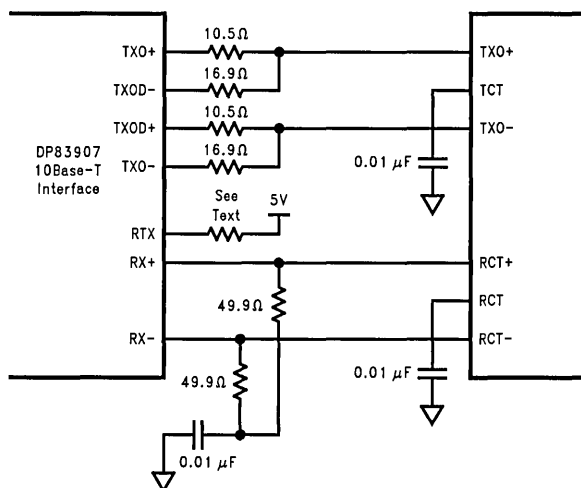
The DP83907 can be used with external Medium Attachment Unit (MAU). The interface is via the 15-pin D-type connector and is a direct interface to the DP83907's ENDEC. The DP83907's ENDEC only requires  $39.2\Omega$  terminations on receive and collision inputs. The transmit outputs do not require any pull-down resistors.

#### 3.9 UTP/STP Function

The TPI Transceiver supports both shielded (STP) and unshielded twisted pair (UTP) cable. UTP is the default but STP can be enabled during configuration by a pull-up resistor on MSA7 or by setting bit D6 high of configuration register C. In UTP mode TXO+ and TXOD+ are driven and TXO- and TXOD- are TRI-STATED. In STP mode TXO- and TXOD- are driven and TXO+ and TXOD+ are TRI-STATED.

#### 3.10 AUTO-SWITCH Function

The auto-switch function can be enabled at configuration by pull-up resistor on MSA5 or by setting bit D4 high of the configuration register C. When auto-switch is enabled link integrity should also be enabled. It allows the transceiver to switch between TP and AUI outputs. If there is an absence of link pulses the transceiver will switch to AUI mode. Similarly when the transceiver starts detecting link pulses it will switch to TP mode. The switching between modes is done after the current packet has been transmitted or received. If the twisted pair output is jabbering and gets into link fail state then the switch to AUI mode is only done after the jabbering is done including the times it takes to unjab (unjab time). When auto-switching is enabled the THIN output is automatically generated if AUI is selected.



TL/F/12335-5

FIGURE 6. Circuitry to connect the DP83907 to Twisted pair cable with internal filter.

## 4.0 SOFTWARE SUPPORT

### 4.1 Configuration and Diagnostics Software

If you are offering a jumper-less solution, the user will require configuration software. The DP83907EB-AT comes with configuration and diagnostics software, called AT5CFG.

AT5CFG will detect and list all the boards installed in the system. The I/O address and interrupt assignment will also be shown. A warning message will appear if there is any conflict in resource allocation.

There are 2 options of saving a changed configuration. Saving a configuration temporarily will render the EEPROM unmodified. Once the system is powered up again the configuration will default to the original settings. Saving the configuration to the EEPROM will permanently store the configuration until another modification is made. The software enable bit (bit 7 of the configuration register C) must be zero for changing configuration register A, B and C by software.

The adapter initialization option will check to see whether a configurable board is present. If so, its configuration parameters will be read and displayed. Otherwise, an error message will be displayed. If the board is non-configurable, the user will be alerted by a warning message. The diagnostic option will do a simple test on cache, interrupt and buffer memory.

The advanced network diagnostic, "Ping Pong Test", can be run only after the board is initialized and passes the diagnostic test described above. Two stations are needed, one master and one slave. In order to test both transmit and receive logic, each station should be tested as a master and slave.

### 4.2 EEPROM in-situ Programming

The software called EEPROM is used to program the Evaluation Board's EEPROM in-situ. There is also a data file holding previous programmed data to facilitate repeated programming. An error message will be prompted if there is no DP83907 present or the programming in-situ feature has been disabled.

### 4.3 Driver Software

The DP83907 can use all NE2000 drivers and NE2000plus drivers in I/O mode only.

## 5.0 REFERENCES

1. DP83907 Data Sheet, National Semiconductor
2. DP83905EB-AT AT/LANTIC Evaluation Board AN875, National Semiconductor
3. DP83905EB-AT AT/LANTIC Hardware Users' Guide AN897, National Semiconductor

# DP83934CVUL-20/25 MHz SONIC™-T

## Systems-Oriented Network Interface Controller with Twisted Pair Interface

### General Description

The SONIC-T (Systems-Oriented Network Interface Controller with Twisted Pair) is a second-generation Ethernet Controller designed to meet the demands of today's high-speed 32- and 16-bit systems. Its system interface operates with a high speed DMA that typically consumes less than 5% of the bus bandwidth. Selectable bus modes provide both big and little endian byte ordering and a clean interface to standard microprocessors. The linked-list buffer management system of SONIC-T offers maximum flexibility in a variety of environments from PC-oriented adapters to high-speed motherboard designs. Furthermore, the SONIC-T integrates a fully-compatible IEEE 802.3 Encoder/Decoder (ENDEC) and a Twisted Pair Interface which provide a one-chip solution for Ethernet when using 10BASE-T. When using 10BASE2 or 10BASE5, the SONIC-T may be paired with the DP8392 Coaxial Transceiver Interface to achieve a simple 2-chip solution.

For increased performance, the SONIC-T implements a unique buffer management scheme to efficiently process receive and transmit packets in system memory. No intermediate packet copy is necessary. The receive buffer management uses three areas in memory for (1) allocating additional resources, (2) indicating status information, and (3) buffering packet data. During reception, the SONIC-T stores packets in the buffer area, then indicates receive status and control information in the descriptor area. The system allocates more memory resources to the SONIC-T by adding

descriptors to the memory resource area. The transmit buffer management uses two areas in memory:

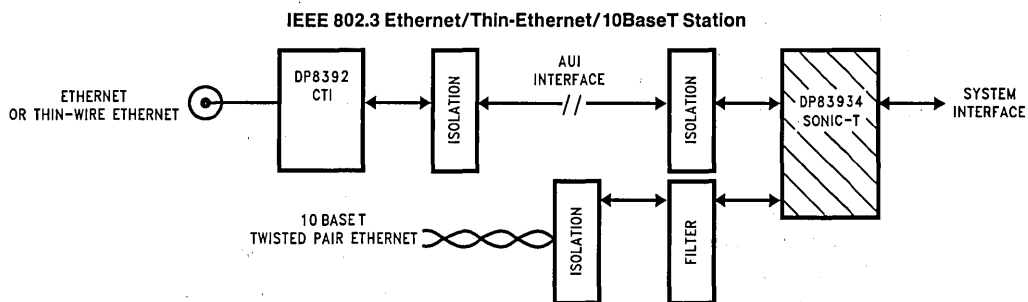
1. indicating status and control information;
2. fetching packet data.

The system can create a transmit queue allowing multiple packets to be transmitted from a single transmit command. The packet data can reside on any arbitrary byte boundary and can exist in several non-contiguous locations.

### Features

- 32-bit non-multiplexed address and data bus
- Auto AUI/TPI selection
- High-speed interruptible DMA
- Linked-list buffer management maximizes flexibility
- Two independent 32-byte transmit and receive FIFOs
- Bus compatibility for all standard microprocessors
- Supports big and little endian formats
- Integrated IEEE 802.3 ENDEC
- Integrated Twisted Pair Interface
- Complete address filtering for up to 16 physical and/or multicast addresses
- 32-bit general-purpose timer
- Loopback diagnostics
- Fabricated in low-power CMOS
- 160 PQFP package
- Full network management facilities support the 802.3 layer management standard
- Integrated support for bridge and repeater applications

### System Diagram



TL/F/11719-1

## Table of Contents

### 1.0 CONNECTION DIAGRAMS

- 1.1 Pin Connection Diagram, National/Intel Mode
- 1.2 Pin Connection Diagram, Motorola Mode

### 2.0 PIN DESCRIPTION

### 3.0 FUNCTIONAL DESCRIPTION

- 3.1 Twisted Pair Interface Module
- 3.2 IEEE 802.3 Encoder/Decoder (ENDEC) Unit
  - 3.2.1 ENDEC Operation
  - 3.2.2 Selecting an External ENDEC
- 3.3 Media Access Control (MAC) Unit
  - 3.3.1 MAC Receive Section
  - 3.3.2 MAC Transmit Section
- 3.4 Data Width and Byte Ordering
- 3.5 FIFO and Control Logic
  - 3.5.1 Receive FIFO
  - 3.5.2 Transmit FIFO
- 3.6 Status and Configuration Registers
- 3.7 Bus Interface
- 3.8 Loopback and Diagnostics
  - 3.8.1 Loopback Procedure
- 3.9 Network Management Functions

### 4.0 TRANSMIT/RECEIVE IEEE 802.3 FRAME FORMAT

- 4.1 Preamble and Start of Frame Delimiter (SFD)
- 4.2 Destination Address
- 4.3 Source Address
- 4.4 Length/Type Field
- 4.5 Data Field
- 4.6 FCS Field
- 4.7 MAC (Media Access Control) Conformance

### 5.0 BUFFER MANAGEMENT

- 5.1 Buffer Management Overview
- 5.2 Descriptor Areas
  - 5.2.1 Naming Convention for Descriptors
  - 5.2.2 Abbreviations
  - 5.2.3 Buffer Management Base Addresses
- 5.3 Descriptor Data Alignment
- 5.4 Receive Buffer Management
  - 5.4.1 Receive Resource Area (RRA)
  - 5.4.2 Receive Buffer Area (RBA)
  - 5.4.3 Receive Descriptor Area (RDA)
  - 5.4.4 Receive Buffer Management Initialization
  - 5.4.5 Beginning of Reception
  - 5.4.6 End of Packet Processing
  - 5.4.7 Overflow Conditions

### 5.5 Transmit Buffer Management

- 5.5.1 Transmit Descriptor Area (TDA)
- 5.5.2 Transmit Buffer Area (TBA)
- 5.5.3 Preparing to Transmit
- 5.5.4 Dynamically Adding TDA Descriptors

### 6.0 SONIC-T REGISTERS

- 6.1 The CAM Unit
  - 6.1.1 The Load CAM Command
- 6.2 Status/Control Registers
- 6.3 Register Description
  - 6.3.1 Command Register
  - 6.3.2 Data Configuration Register
  - 6.3.3 Receive Control Register
  - 6.3.4 Transmit Control Register
  - 6.3.5 Interrupt Mask Register
  - 6.3.6 Interrupt Status Register
  - 6.3.7 Data Configuration Register 2
  - 6.3.8 Transmit Registers
  - 6.3.9 Receive Registers
  - 6.3.10 CAM Registers
  - 6.3.11 Tally Counters
  - 6.3.12 General Purpose Timer
  - 6.3.13 Silicon Revision Register

### 7.0 BUS INTERFACE

- 7.1 Pin Configurations
- 7.2 System Configuration
- 7.3 Bus Operations
  - 7.3.1 Acquiring the Bus
  - 7.3.2 Block Transfers
  - 7.3.3 Bus Status
  - 7.3.4 Bus Mode Compatibility
  - 7.3.5 Master Mode Bus Cycles
  - 7.3.6 Bus Exceptions (Bus Retry)
  - 7.3.7 Slave Mode Bus Cycle
  - 7.3.8 On-Chip Memory Arbiter
  - 7.3.9 Chip Reset

### 8.0 NETWORK INTERFACING

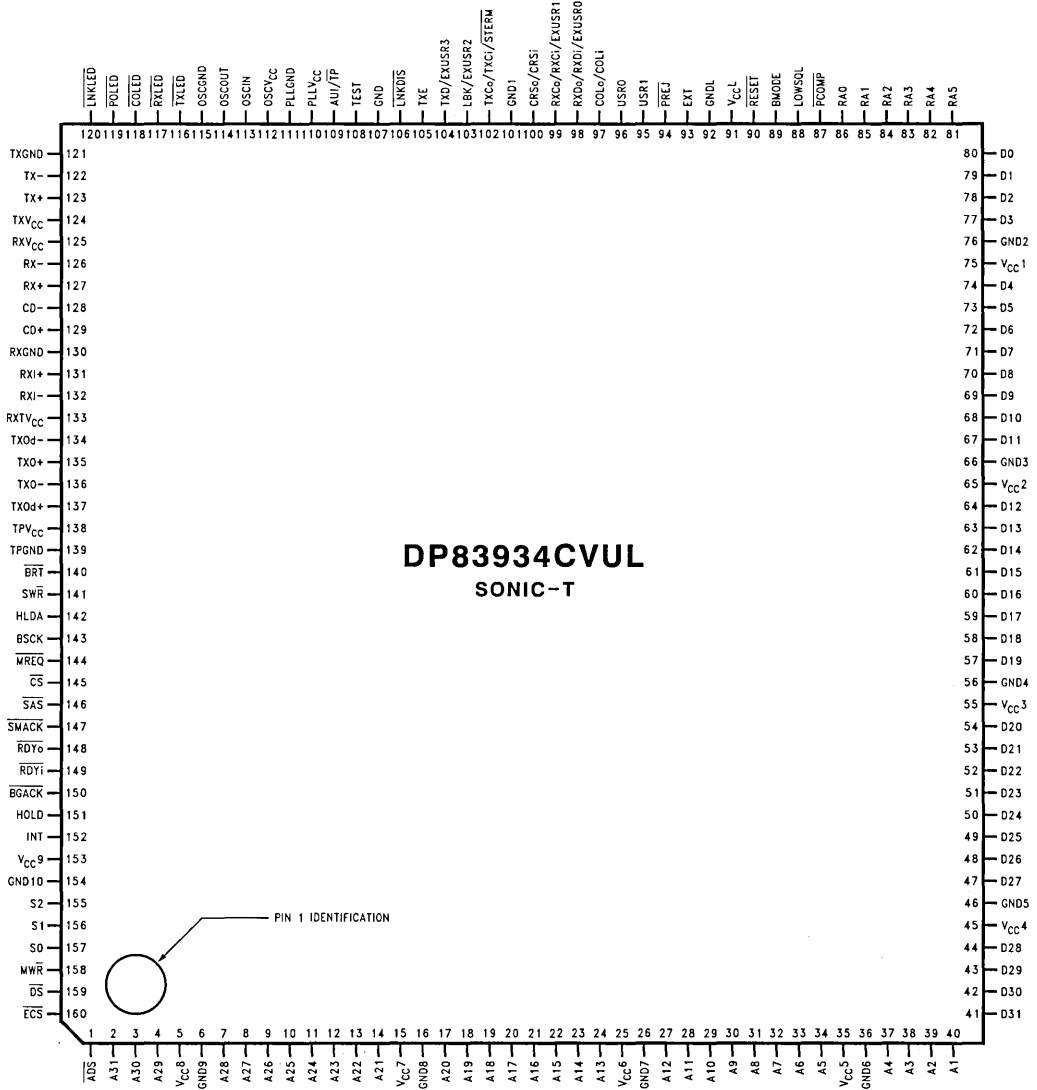
- 8.1 Manchester Encoder and Differential Driver
  - 8.1.1 Manchester Decoder
  - 8.1.2 Collision Translator
  - 8.1.3 Oscillator Inputs
  - 8.1.4 Power Supply Considerations
- 8.2 Twisted Pair Interface Module

### 9.0 AC AND DC SPECIFICATIONS

### 10.0 AC TIMING TEST CONDITIONS

# 1.0 Connection Diagrams

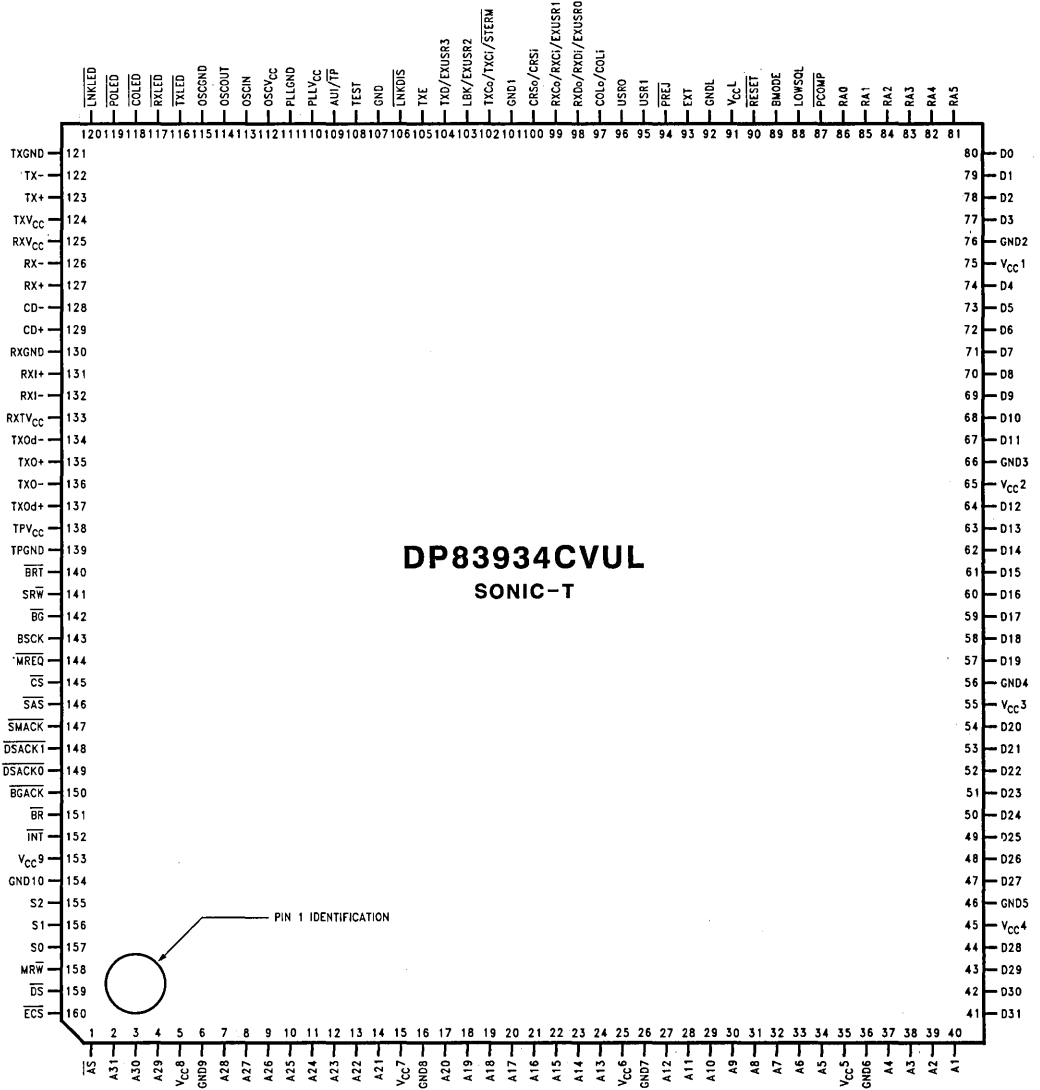
## 1.1 PIN CONNECTION DIAGRAM, NATIONAL/INTEL MODE



TL/F/11719-2

# 1.0 Connection Diagrams (Continued)

## 1.2 PIN CONNECTION DIAGRAM, MOTOROLA MODE





## 2.0 Pin Description

I = Input  
 O = Output  
 Z = TRI-STATE® Input, TTL compatible  
 ECL = Emitter Coupled Logic type drivers for interfacing to the Attachment Unit Interface.  
 TP = Totem Pole type drivers. These drivers are driven either high or low and are always driven. Drive levels are CMOS compatible.

TRI = TRI-STATE drivers. These pins are driven high, low or TRI-STATE. Drive levels are CMOS compatible. These pins may also be inputs (depending on the pin).  
 OC = Open Collector type drivers. These drivers are TRI-STATE when inactive and are driven low when active. These pins may also be inputs (depending on the pin).  
 TPI = Twisted Pair Interface.  
 Pin names which contain a "/" indicate dual function pins.

TABLE 2-1. Pin Description

Symbol	Driver Type	Direction	Description
<b>NETWORK INTERFACE PINS</b>			
EXT		I	<b>EXTERNAL ENDEC SELECT:</b> Tying this pin to $V_{CC}$ (EXT = 1) disables the internal ENDEC and allows an external ENDEC to be used. Tying this pin to ground (EXT = 0) enables the internal ENDEC. This pin must be tied either to $V_{CC}$ or ground. Note the alternate pin definitions for CRS <sub>o</sub> /CRS <sub>i</sub> , COL <sub>o</sub> /COL <sub>i</sub> , RXD <sub>o</sub> /RXD <sub>i</sub> , RXCo/RXC <sub>i</sub> , and TXCo/TXC <sub>i</sub> . When EXT = 0 the first pin definition is used and when EXT = 1 the second pin definition is used.
AUI/TP		I	<b>ATTACHMENT UNIT INTERFACE (AUI)/TWISTED PAIR (TP) SELECT:</b> Tying this pin to $V_{CC}$ (AUI/TP = 1) enables the AUI mode for interface with the ENDEC unit. Tying this pin to GND (AUI/TP = 0) enables the TPI Module mode for interface with the ENDEC unit.
TXO <sub>d+</sub> , TXO <sub>+</sub> , TXO <sub>-</sub> , TXO <sub>d-</sub>	TPI	O	<b>TWISTED PAIR TRANSMIT OUTPUTS:</b> These high drive CMOS level outputs are resistively combined external to the chip to produce a differential output signal with equalization to compensate for Intersymbol Interference (ISI) on the twisted pair medium.
RXI <sub>+</sub> , RXI <sub>-</sub>	TPI	I	<b>TWISTED PAIR RECEIVE INPUTS:</b> These inputs feed a differential amplifier which passes valid data to the ENDEC module.
TXLED	TP	O	<b>TRANSMIT:</b> An active low output. It is asserted for approximately 50 ms whenever the SONIC-T Controller transmits data in either AUI or TPI modes.
RXLED	TP	O	<b>RECEIVE:</b> An active low output. It is asserted for approximately 50 ms whenever receive data is detected in either AUI or TPI mode.
COLED	TP	O	<b>COLLISION:</b> An active low output. It is asserted for approximately 50 ms whenever the SONIC-T Controller detects a collision in either AUI or TPI modes.
POLED	TP	O	<b>POLARITY:</b> An active low output. This signal is normally inactive. When the TPI module detects seven consecutive link pulses or three consecutive received packets with reversed polarity, it is asserted.
LNKLED	TP	O	<b>GOOD LINK:</b> An active low output. This pin operates as an output to display link integrity status if this function has not been disabled by the LNKDIS pin described below. This output is off if the SONIC-T Controller is in AUI mode or if link testing is enabled and the link integrity is bad (i.e., the twisted pair link has been broken). This output is on if the SONIC-T Controller is in Twisted Pair Interface (TPI) mode, link integrity checking is enabled and the link integrity is good (i.e., the twisted pair link has not been broken) or if the link testing is disabled.
LNKDIS		I	<b>LINK DISABLE:</b> When this pin is tied to GND (LNKDIS = 0), the link test pulse generation and integrity checking function are both disabled.
LOWSQL		I	<b>LOW SQUELCH SELECT:</b> Tying this pin to $V_{CC}$ (LOWSQL = 1) sets the squelch mode to use a squelch threshold level lower than that of the 10BASE-T specification (see Section 3.1).

## 2.0 Pin Description (Continued)

TABLE 2-1. Pin Description (Continued)

Symbol	Driver Type	Direction	Description
<b>NETWORK INTERFACE PINS (Continued)</b>			
CD+		I	<b>AUI COLLISION +</b> : The positive differential collision input from the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
CD-		I	<b>AUI COLLISION -</b> : The negative differential collision input from the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
RX+		I	<b>AUI RECEIVE +</b> : The positive differential receive data input from the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
RX-		I	<b>AUI RECEIVE -</b> : The negative differential receive data input from the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
TX+	ECL	O	<b>AUI TRANSMIT +</b> : The positive differential transmit output to the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
TX-	ECL	O	<b>AUI TRANSMIT -</b> : The negative differential transmit output to the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
CRSo/ CRSi	TP	O I	<b>CARRIER SENSE OUTPUT (CRSo)</b> from the internal ENDEC (EXT = 0): When EXT = 0 the CRSo signal is internally connected between the ENDEC and MAC units. It is asserted on the first valid high-to-low transition in the receive data (RX $\pm$ ). This signal remains active 1.5 bit times after the last bit of data. Although this signal is used internally by the SONIC-T, it is also provided as an output to the user. <b>CARRIER SENSE INPUT (CRSi)</b> from an external ENDEC (EXT = 1): The CRSi signal is activated high when the external ENDEC detects valid data at its receive inputs.
COLo/ COLi	TP	O I	<b>COLLISION OUTPUT (COLo)</b> from the internal ENDEC (EXT = 0): When EXT = 0 the COLo signal is internally connected between the ENDEC and MAC units. This signal generates an active high signal when the 10 MHz collision signal from the transceiver is detected. Although this signal is used internally by the SONIC-T, it is also provided as an output to the user. <b>COLLISION DETECT INPUT (COLi)</b> from an external ENDEC (EXT = 1): The COLi signal is activated from an external ENDEC when a collision is detected. This pin is monitored during transmissions from the beginning of the Start of Frame Delimiter (SFD) to the end of the packet. At the end of transmission, this signal is monitored by the SONIC-T for CD heartbeat.
RXDo/ RXDi/ EXUSR0	TP TRI	O I O, Z	This pin will be TRI-STATE until the DCR has been written to. (See Section 6.3.2, EXBUS, for more information.) <b>RECEIVE DATA OUTPUT (RXDo)</b> from the internal ENDEC (EXT = 0): NRZ data output. When EXT = 0 the RXDo signal is internally connected between the ENDEC and MAC units. This signal must be sampled on the rising edge of the receive clock output (RXCo). Although this signal is used internally by the SONIC-T, it is also provided as an output to the user. <b>RECEIVE DATA INPUT (RXDi)</b> from an external ENDEC (EXT = 1): The NRZ data decoded from the external ENDEC. This data is clocked in on the rising edge of RXCi. <b>EXTENDED USER OUTPUT (EXUSR0)</b> : When EXBUS has been set (see Section 6.3.2), this pin becomes a programmable output. It will remain TRI-STATE until the SONIC-T becomes a bus master, at which time it will be driven according to the value programmed in the DCR2 (see Section 6.3.7).
RXCo/ RXCi/ EXUSR1	TP TRI	O I O, Z	This pin will be TRI-STATE until the DCR has been written to. (See Section 6.3.2, EXBUS, for more information.) <b>RECEIVE CLOCK OUTPUT (RXCo)</b> from the internal ENDEC (EXT = 0): When EXT = 0 the RXCo signal is internally connected between the ENDEC and MAC units. This signal is the receive clock that is derived from the Manchester data stream. It remains active 5-bit times after the deassertion of CRSo. Although this signal is used internally by the SONIC-T it is also provided as an output to the user. <b>RECEIVE CLOCK INPUT (RXCi)</b> from an external ENDEC (EXT = 1): The receive clock that is derived from the Manchester data stream. This signal is generated from an external ENDEC. <b>EXTENDED USER OUTPUT (EXUSR1)</b> : When EXBUS has been set (see Section 6.3.2), this pin becomes a programmable output. It will remain TRI-STATE until the SONIC-T becomes a bus master, at which time it will be driven according to the value programmed in the DCR2 (see Section 6.3.7).

## 2.0 Pin Description (Continued)

TABLE 2-1. Pin Description (Continued)

Symbol	Driver Type	Direction	Description
<b>NETWORK INTERFACE PINS (Continued)</b>			
TXD/ EXUSR3	TP TRI	O O, Z	This pin will be TRI-STATE until the DCR has been written to. (See Section 6.3.2, EXBUS, for more information.) <b>TRANSMIT DATA (TXD):</b> The serial NRZ data from the MAC unit which is to be decoded by an external ENDEC. Data is valid on the rising edge of TXC. Although this signal is used internally by the SONIC-T it is also provided as an output to the user. <b>EXTENDED USER OUTPUT (EXUSR3):</b> When EXBUS has been set (see Section 6.3.2), this pin becomes a programmable output. It will remain TRI-STATE until the SONIC-T becomes a bus master, at which time it will be driven according to the value programmed in the DCR2 (see Section 6.3.7).
TXE	TP	O	<b>TRANSMIT ENABLE:</b> This pin is driven high when the SONIC-T begins transmission and remains active until the last byte is transmitted. Although this signal is used internally by the SONIC-T it is also provided as an output to the user.
TXCo/ TXCi/ STERM	TRI	O, Z I I	This pin will be TRI-STATE until the DCR has been written to. (See Section 6.3.2, EXBUS, for more information.) <b>TRANSMIT CLOCK OUTPUT (TXCo)</b> from the internal ENDEC (EXT = 0): This 10 MHz transmit clock output is derived from the 20 MHz oscillator input. When EXT = 0 the TXCo signal is internally connected between the ENDEC and MAC units. Although this signal is used internally by the SONIC-T, it is also provided as an output to the user. <b>TRANSMIT CLOCK INPUT (TXCi)</b> from an external ENDEC (EXT = 1): This input clock from an external ENDEC is used for shifting data out of the MAC unit serializer. This clock is nominally 10 MHz. <b>SYNCHRONOUS TERMINATION (STERM):</b> When the SONIC-T is a bus master, it samples this pin before terminating its memory cycle. This pin is sampled synchronously and may only be used in asynchronous bus mode when BMODE = 1. (See Section 7.2.5 for more details.)
LBK/ EXUSR2	TP TRI	O O, Z	This pin will be TRI-STATE until the DCR has been written to. (See Section 6.3.2, EXBUS, for more information.) <b>LOOPBACK (LBK):</b> When ENDEC Loopback mode is enabled, LBK is asserted high. Although this signal is used internally by the SONIC-T it is also provided as an output to the user. <b>EXTENDED USER OUTPUT (EXUSR2):</b> When EXBUS has been set (see Section 6.3.2), this pin becomes a programmable output. It will remain TRI-STATE until the SONIC-T becomes a bus master, at which time it will be driven according to the value programmed in the DCR2 (see Section 6.3.7).
PCOMP	TRI	O, Z	<b>PACKET COMPRESSION:</b> This pin is used with the Management Bus of the DP83950, Repeater Interface Controller (RIC). The SONIC-T can be programmed to assert PCOMP whenever there is a CAM match, or when there is not a match. The RIC uses this signal to compress (shorten) a received packet for management purposes and to reduce memory usage. (See the DP83950 datasheet for more details on the RIC Management Bus.) The operation of this pin is controlled by bits 1 and 2 in the DCR2 register. PCOMP will remain TRI-STATE until these bits are written to. This signal is asserted right after the 4th bit of the 7th byte of the incoming packet and is deasserted one transmit clock (TXC) after CSR is driven low.
PREJ		I	<b>PACKET REJECT:</b> This signal is used to reject received packets. When asserted low for at least two receive clock cycles (RXC), the SONIC-T will reject the incoming packet. This pin can be asserted up to the 2nd to the last bit of reception to reject a packet.

## 2.0 Pin Description (Continued)

TABLE 2-1. Pin Description (Continued)

Symbol	Driver Type	Direction	Description
<b>NETWORK INTERFACE PINS (Continued)</b>			
OSCIN		I	<b>CRYSTAL FEEDBACK INPUT OR EXTERNAL OSCILLATOR INPUT:</b> This signal is used to provide clocking signals for the internal ENDEC. A crystal may be connected to this pin along with OSCOUT, or an oscillator module may be used. See Section 8.1.3 for more information about using an oscillator or crystal.
OSCOUT	TP	O	<b>CRYSTAL FEEDBACK OUTPUT:</b> This signal is used to provide clocking signals for the internal ENDEC. A crystal can be connected to this pin along with OSCIN. See Section 8.1.3 for more information about using an oscillator or crystal.
<b>BUS INTERFACE PINS (BOTH BUS MODES)</b>			
BMODE		I	<b>BUS MODE:</b> This input enables the SONIC-T to be compatible with standard microprocessor buses. The level of this pin affects byte ordering (little or big endian) and controls the operation of the bus interface control signals. A high level (tied to $V_{CC}$ ) selects Motorola mode (big endian) and a low level (tied to ground) selects National/Intel mode (little endian). Note the alternate pin definitions for $\overline{AS}/ADS$ , $M\overline{R}\overline{W}/MWR$ , $\overline{INT}/INT$ , $\overline{B}\overline{R}/HOLD$ , $\overline{B}\overline{G}/HLDA$ , $S\overline{R}\overline{W}/SWR$ , $D\overline{SACK}0/\overline{RDY}0$ , and $D\overline{SACK}1/\overline{RDY}1$ . (See Sections 7.3.1, 7.3.4, and 7.3.5 for bus interface information.)
D31–D0	TRI	I, O, Z	<b>DATA BUS:</b> These bidirectional lines are used to transfer data on the system bus. When the SONIC-T is a bus master, 16-bit data is transferred on D15–D0 and 32-bit data is transferred on D31–D0. When the SONIC-T is accessed as a slave, register data is driven onto line D15–D0. D31–D16 are held TRI-STATE.
A31–A1	TRI	O, Z	<b>ADDRESS BUS:</b> These signals are used by the SONIC-T to drive the DMA address after the SONIC-T has acquired the bus. Since the SONIC-T aligns data to word boundaries, only 31 address lines are needed.
RA5–RA0		I	<b>REGISTER ADDRESS BUS:</b> These signals are used to access SONIC-T's internal registers. When the SONIC-T is accessed, the CPU drives these lines to select the desired SONIC-T register.
$\overline{RESET}$		I	<b>RESET:</b> This signal is used to hardware reset the SONIC-T. When asserted low, the SONIC-T transitions into the reset state after 10 transmit clocks or 10 bus clocks if the bus clock period is greater than the transmit clock period.
S2–S0	TP	O	<b>BUS STATUS:</b> These three signals provide a continuous status of the current SONIC-T bus operations. See Section 7.3.3 for status definitions.
BCLK		I	<b>BUS CLOCK:</b> This clock provides the timing for the SONIC-T DMA engine.
$\overline{CS}$		I	<b>CHIP SELECT:</b> The system asserts this pin low to access the SONIC-T's registers. The registers are selected by placing an address on lines RA5–RA0. <b>Note:</b> Both $\overline{CS}$ and $\overline{MREQ}$ must not be asserted concurrently. If these signals are successively asserted, there must be at least two bus clocks between the deasserting edge of the first signal and the asserting edge of the second signal.

## 2.0 Pin Description (Continued)

TABLE 2-1. Pin Description (Continued)

Symbol	Driver Type	Direction	Description
<b>BUS INTERFACE PINS (BOTH BUS MODES) (Continued)</b>			
$\overline{SAS}$		I	<b>SLAVE ADDRESS STROBE:</b> The system asserts this pin to latch the register address on lines RA0–RA5.
$\overline{DS}$	TRI	O, Z	<b>DATA STROBE:</b> When the SONIC-T is bus master, it drives this pin low during a read cycle to indicate that the slave device may drive data onto the bus; in a write cycle, this pin indicates that the SONIC-T has placed valid data onto the bus.
$\overline{BRT}$		I	<b>BUS RETRY:</b> When the SONIC-T is bus master, the system asserts this signal to rectify a potentially correctable bus error. This pin has two modes. Mode 1 (the LBR in the Data Configuration Register is set to 0): Assertion of this pin forces the SONIC-T to terminate the current bus cycle and will repeat the same cycle after $\overline{BRT}$ has been deasserted. Mode 2 (the LBR bit in the Data Configuration register is set to 1): Assertion of this signal forces the SONIC-T to retry the bus operation as in Mode 1. However, the SONIC-T will not continue DMA operations until the BR bit in the ISR is reset.
$\overline{ECS}$	TRI	O, Z	<b>EARLY CYCLE START:</b> This output gives the system earliest indication that a memory operation is occurring. This signal is driven low at the rising edge of T1 and high at the falling edge of T1.
<b>SHARED-MEMORY ACCESS PINS</b>			
$\overline{MREQ}$		I	<b>MEMORY REQUEST:</b> The system asserts this signal low when it attempts to access the shared-buffer RAM. The on-chip arbiter resolves accesses between the system and the SONIC-T. <b>Note:</b> Both $\overline{CS}$ and $\overline{MREQ}$ must not be asserted concurrently. If these signals are successively asserted, there must be at least two bus clocks between the deasserting edge of the first signal and the asserting edge of the second signal.
$\overline{SMACK}$	TP	O	<b>SLAVE AND MEMORY ACKNOWLEDGE:</b> SONIC-T asserts this dual function pin low in response to either a Chip Select ( $\overline{CS}$ ) or a Memory Request ( $\overline{MREQ}$ ) when the SONIC-T's registers or its buffer memory is available for accessing. This pin can be used for enabling bus drivers for dual-bus systems.
<b>BUS INTERFACE PINS (NATIONAL/INTEL MODE, BMODE = 0)</b>			
$\overline{ADS}$	TRI	O, Z	<b>ADDRESS STROBE (<math>\overline{ADS}</math>):</b> The rising edge indicates valid status and address.
$\overline{MWR}$	TRI	O, Z	<b>MEMORY WRITE/READ STROBE <math>\overline{MWR}</math>:</b> When the SONIC-T has acquired the bus, this signal indicates the direction of the data transfer. The signal is low during a read cycle and high during a write cycle.
$\overline{INT}$	TP	O	<b>INTERRUPT (<math>\overline{INT}</math>):</b> Indicates that an interrupt (if enabled) is pending from one of the sources indicated by the Interrupt Status register. Interrupts that are disabled in the Interrupt Mask register will not activate this signal.
$\overline{HOLD}$	TP	O	<b>HOLD REQUEST (<math>\overline{HOLD}</math>):</b> The SONIC-T drives this pin high when it intends to use the bus and is driven low when inactive.
$\overline{HLDA}$		I	<b>HOLD ACKNOWLEDGE (<math>\overline{HLDA}</math>):</b> This signal is used to inform the SONIC-T that it has attained the bus. When the system asserts this pin high, the SONIC-T has gained ownership of the bus.
$\overline{BGACK}$	TRI	O, Z	<b>BUS GRANT ACKNOWLEDGE:</b> This pin is only used when BMODE = 1.
$\overline{SWR}$		I	<b>SLAVE READ/WRITE STROBE (<math>\overline{SWR}</math>):</b> The system asserts this pin to indicate whether it will read from or write to the SONIC-T's registers. This signal is asserted low during a read and high during a write.
$\overline{RDYi}$		I	<b>READY INPUT (<math>\overline{RDYi}</math>, BMODE = 0):</b> When the SONIC-T is a bus master, the system asserts this signal high to insert wait-states and low to terminate the memory cycle. This signal is sampled synchronously or asynchronously depending on the state of the SBUS bit. (See Sections 7.3.5 and 6.3.2 for details.)
$\overline{RDYo}$	TP	O	<b>READY OUTPUT (<math>\overline{RDYo}</math>):</b> When a register is accessed, the SONIC-T asserts this signal to terminate the slave cycle.

## 2.0 Pin Description (Continued)

TABLE 2-1. Pin Description (Continued)

Symbol	Driver Type	Direction	Description
<b>BUS INTERFACE PINS (MOTOROLA MODE, BMODE = 1)</b>			
$\overline{AS}$	TRI	O, Z	<b>ADDRESS STROBE (<math>\overline{AS}</math>):</b> The falling edge indicates valid status and address. The rising edge indicates the termination of the memory cycle.
$\overline{MRW}$	TRI	O, Z	<b>MEMORY READ/WRITE STROBE (<math>\overline{MRW}</math>):</b> When the SONIC-T has acquired the bus, this signal indicates the direction of the data transfer. This signal is high during a read cycle and low during a write cycle.
$\overline{INT}$	OC	O, Z	<b>INTERRUPT (<math>\overline{INT}</math>):</b> Indicates that an interrupt (if enabled) is pending from one of the sources indicated by the Interrupt Status register. Interrupts that are disabled in the Interrupt Mask register will not activate this signal.
$\overline{BR}$	OC	O, Z	<b>BUS REQUEST (<math>\overline{BR}</math>):</b> The SONIC-T asserts this pin low when it attempts to gain access to the bus. When inactive this signal is at TRI-STATE.
$\overline{BG}$		I	<b>BUS GRANT (<math>\overline{BG}</math>):</b> This signal is a bus grant. The system asserts this pin low to indicate potential mastership of the bus.
$\overline{BGACK}$	TRI	O, Z	<b>BUS GRANT ACKNOWLEDGE:</b> The SONIC-T asserts this pin low when it has determined that it can gain ownership of the bus. The SONIC-T checks the following conditions before driving $\overline{BGACK}$ : <ol style="list-style-type: none"> <li><math>\overline{BG}</math> has been received through the bus arbitration process.</li> <li><math>\overline{AS}</math> is deasserted, indicating that the previous master has finished using the bus.</li> <li><math>\overline{DSACK0}</math> and <math>\overline{DSACK1}</math> are deasserted, indicating that the previous slave device is off the bus.</li> <li><math>\overline{BGACK}</math> is deasserted, indicating that the previous master is off the bus.</li> </ol>
$\overline{SRW}$		I	<b>SLAVE READ/WRITE (<math>\overline{SRW}</math>):</b> The system asserts this pin to indicate whether it will read from or write to the SONIC-T's registers. This signal is asserted high during a read and low during a write.
$\overline{DSACK0}$ $\overline{DSACK1}$	TRI TRI	I, O, Z I, O, Z	<b>DATA AND SIZE ACKNOWLEDGE 0 AND 1 (<math>\overline{DSACK0,1}</math> BMODE = 1):</b> These pins are the output slave acknowledge to the system when the SONIC-T registers have been accessed and the input slave acknowledgement when the SONIC-T is busmaster. When a register has been accessed, the SONIC-T drives both $\overline{DSACK0}$ and $\overline{DSACK1}$ pins low to terminate the slave cycle. (Note that the SONIC-T responds as a 32-bit peripheral by driving both $\overline{DSACK0}$ and $\overline{DSACK1}$ low, but drives data only on lines D0–D15. Lines D16–D32 are driven, but invalid.) When the SONIC-T is bus master, it samples these pins before terminating its memory cycle. When SONIC-T is in 32-bit bus master mode, both $\overline{DSACK0}$ and $\overline{DSACK1}$ must be asserted to terminate the cycle. However, if the SONIC-T is in 16-bit bus master mode, only the assertion of $\overline{DSACK1}$ is required to terminate the cycle. These pins are sampled synchronously or asynchronously depending on the state of the SBUS bit in the Data Configuration register. (See Section 7.3.5 for details.) Note that the SONIC-T does not allow dynamic bus sizing. Bus size is statically defined in the Data Configuration register (see Section 6.3.2).
<b>USER DEFINABLE PINS</b>			
USR0,1	TRI	I, O, Z	<b>USER DEFINE 0,1:</b> These signals are inputs when the SONIC-T is hardware reset and are outputs when the SONIC-T is a bus master (HLDA or $\overline{BGACK}$ asserted). When hard reset ( $\overline{RST}$ ) is low, these signals input directly into bits 8 and 9 of the Data Configuration Register (DCR) respectively. The levels on these pins are latched on the rising edge of $\overline{RST}$ . During busmaster operations (HLDA or $\overline{BGACK}$ is active), these pins are outputs whose levels are programmable through bits 11 and 12 of the DCR respectively. The USR0,1 pins should be pulled up to $V_{CC}$ or pulled down to ground. A 4.7 k $\Omega$ pull-up resistor is recommended.

**2.0 Pin Description** (Continued)

**TABLE 2-1. Pin Description** (Continued)

Symbol	Driver Type	Direction	Description
<b>UNCONNECTED PINS</b>			
TEST		I	<b>FACTORY TEST INPUT:</b> Used to check the chip's internal functions. This pin should be left unconnected during normal operation.
<b>POWER AND GROUND PINS</b>			
V <sub>CC</sub> 1–9 V <sub>CCL</sub>			<b>POWER:</b> The +5V power supply for the digital portions of the SONIC-T.
TXV <sub>CC</sub> RXV <sub>CC</sub> PLL <sub>VCC</sub> OSCV <sub>CC</sub>			<b>POWER:</b> These pins are the +5V power supply for the SONIC-T ENDEC unit. These pins must be tied to V <sub>CC</sub> even if the internal ENDEC is not used.
RXTV <sub>CC</sub> TPV <sub>CC</sub>			<b>POWER:</b> These pins are the +5V power supply for the SONIC-T TPI unit. These pins must be tied to V <sub>CC</sub> even if the internal TPI module is not used.
GND 1–10 GNDL GND			<b>GROUND:</b> These pins are the ground references for the digital portions of the SONIC-T.
TXGND RXGND PLL <sub>GND</sub> OSCGND TPGND			<b>GROUND:</b> These pins are the ground references for the SONIC-T ENDEC unit and TPI module. These pins must be tied to ground even if the internal ENDEC unit and/or the TPI module are not used.

### 3.0 Functional Description

The SONIC-T (Figure 3-1) consists of a twisted pair interface (TPI) module, an encoder/decoder (ENDEC) unit, a media access control (MAC) unit, separate receive and transmit FIFOs, a system buffer management engine, and a user programmable system bus interface unit on a single chip. SONIC-T is highly pipelined providing maximum system level performance. This section provides a functional overview of the SONIC-T.

#### 3.1 TWISTED PAIR INTERFACE MODULE

The TPI consists of five main logic functions:

- a. the Smart Squelch, which determines when valid data is present on the differential receive inputs (RXI±),
- b. the Collision Detector, which checks for simultaneous transmission and reception of data on the differential transmit output (TXO±) and differential receive input (RXI±) pins,
- c. the Link Detector/Generator, which checks the integrity of the cable connecting the two twisted pair modules,

- d. the Jabber, which disables the transmitter if it attempts to transmit a longer than legal packet, and
- e. the Transmitter, which utilizes a Transmit Driver and a Pre-emphasis to transmit Manchester encoded data to the twisted pair network via summing resistors and a transformer/filter.

**Smart Squelch:** The SONIC-T Controller implements an intelligent receive squelch on the RXI± differential inputs to ensure that impulse noise on the receive inputs will not be mistaken for a valid signal.

The squelch circuitry employs a combination of amplitude and timing measurements to determine the validity of data on the twisted pair inputs. There are two voltage level options for the smart squelch. One mode, 10BASE-T mode (Figure 3-2), uses levels that meet the 10BASE-T specification. The second mode, reduced squelch mode, uses a lower squelch threshold level, and can be used in longer cable applications where smaller signal levels may be applied. The squelch level mode can be selected using the LOWSQL input pin (see Section 2.0).

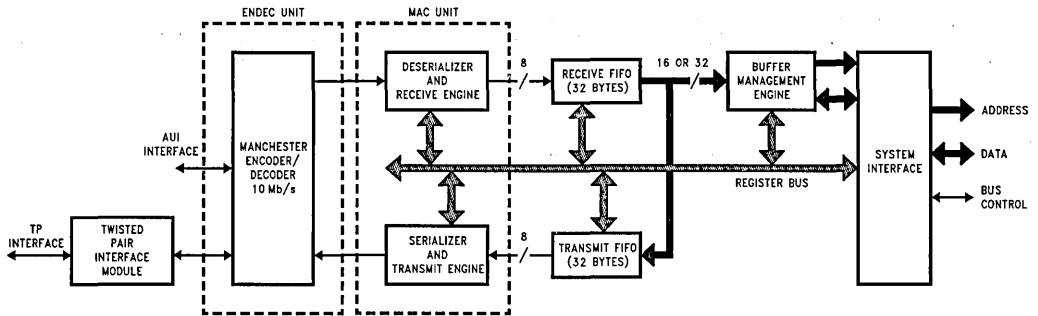


FIGURE 3-1. SONIC-T Block Diagram

TL/F/11719-4

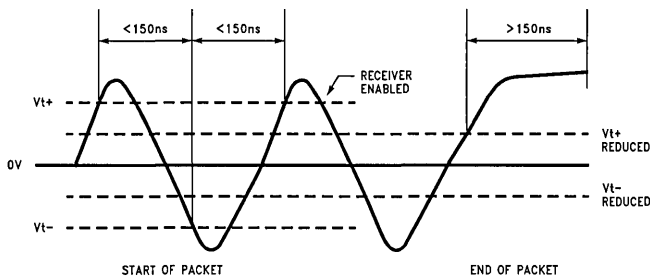


FIGURE 3-2. Twisted Pair Squelch Waveform (10BASE-T Mode)

TL/F/11719-5



### 3.0 Functional Description (Continued)

The signal at the start of the packet is checked by the smart squelch, and any pulses not exceeding the squelch level (either positive or negative, depending upon polarity) will be rejected. Once this first squelch level is overcome correctly, the opposite squelch level must then be exceeded within 150 ns. Finally, the signal must exceed the original squelch level within the next 150 ns time period to ensure that the input waveform will not be rejected. The checking procedure typically results in the loss of three bits at the beginning of each packet.

Only after all these conditions have been satisfied will a control signal be generated to indicate to the remainder of the circuitry that valid data is present. At this time the smart squelch circuitry is reset.

In the reduced squelch mode the operation is identical except that the lower squelch levels shown in *Figure 3-2* are used.

Valid data is considered to be present until either squelch level has not been generated for a time period of more than 150 ns indicating the End of Packet. Once good data has been detected, the squelch levels are reduced to minimize the effect of noise causing premature End of Packet detection.

**Collision:** A collision is detected by the TPI module when the receive and transmit channels are simultaneously active. If the TPI is receiving when a collision is detected it is reported to the controller immediately. If, however, the TPI is transmitting when a collision is detected, the collision is not reported until seven bits have been received while in the collision state. This prevents a collision being reported incorrectly due to noise on the network. The signal to the controller remains for the duration of the collision.

Approximately 1  $\mu$ s after the transmission of each packet, a signal called the Signal Quality Error (SQE) is generated which typically consists of 10 cycles of a 10 MHz signal. This 10 MHz signal, also called the Heartbeat, ensures the continued functioning of the collision circuitry.

**Link Detector/Generator:** The link generator is a timer circuit that generates a link pulse, produced by the transmitter section, as defined by the 10BASE-T specification. The 100 ns wide pulse is transmitted on the TXO+ output every 16 ms in the absence of transmit data.

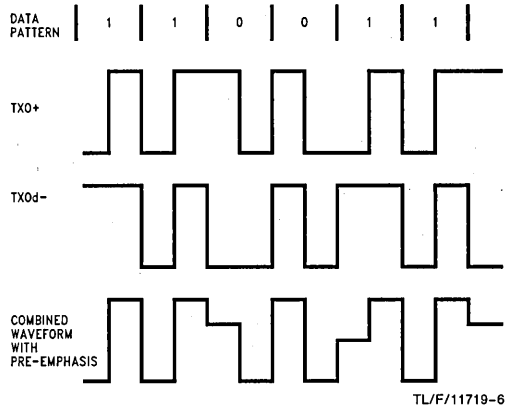
This link pulse is used to check the integrity of the connection to the remote MAU. The link detection circuit checks for valid pulses that are received from the remote unit. If valid link pulses are not received, the link detector will disable the transmit, receive, and collision detection functions.

The LINKLED output can directly drive a LED to show that there is a good twisted pair link. For normal conditions the LED will be on. The link integrity function can be disabled by asserting the LNKDIS input pin.

**Jabber:** The jabber timer monitors the transmitter and disables the transmission if the transmitter is active for greater than 26 ms. The transmitter is then disabled for the whole time that the ENDEC module's internal transmit enable is asserted. This signal has to be deasserted for approximately 750 ms (the unjab time) before the Jabber re-enables the transmit outputs.

**Transmitter:** The transmitter consists of four signals, the true and complement Manchester encoded data (TXO $\pm$ ) and these signals delayed by 50 ns (TXOd $\pm$ ).

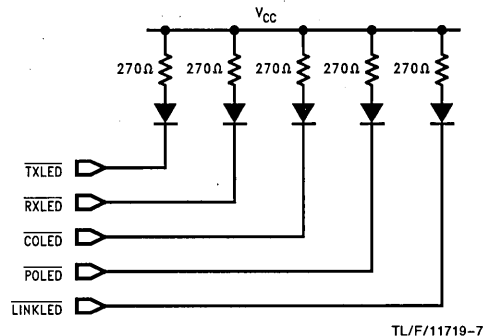
These four signals are resistively combined (see Section 8.2), TXO+ with TXOd- and TXO- with TXOd+, in a configuration referred to as pre-emphasis. This digital pre-emphasis is required to compensate for the low-pass filter effects of the twisted pair cable which causes greater attenuation to the 10 MHz (50 ns) pulses of the Manchester encoded waveform than the 5 MHz (100 ns) pulses.



**FIGURE 3-3. Typical Summed Transmit Waveform**

The signal with pre-emphasis is generated by resistively combining TXO+ and TXOd- (*Figure 3-3*). This signal along with its complement is passed to the transmit filter.

**Status Information:** Status information is provided by the SONIC-T Controller on the RXLED, TXLED, COLED, LINKLED, and POLED outputs as described in the pin description table. These outputs (*Figure 3-4*) are suitable for driving status LEDs.



**FIGURE 3-4. Typical SONIC-T LED Connection**

## 3.0 Functional Description (Continued)

### 3.2 IEEE 802.3 ENCODER/DECODER (ENDEC) UNIT

The Encoder/Decoder (ENDEC) unit is the interface between either the Twisted Pair Interface Module or the Ethernet transceiver and the Media Access Control (MAC) unit. Providing the Manchester data encoding and decoding functions for IEEE 802.3 Ethernet, Thin-Ethernet, or Twisted Pair types of local area networks, the ENDEC operations of SONIC-T are identical to those of the DP83910A CMOS Serial Network Interface device. During transmission, the ENDEC unit combines non-return-zero (NRZ) data from the MAC section and clock pulses to produce Manchester data and sends the converted data differentially to the transceiver. Conversely, during reception, an analog Phase Lock Loop (PLL) decodes the Manchester data into both NRZ formatted data and a receive clock. The SONIC-T ENDEC unit is a functionally complete Manchester encoder/decoder incorporating a balanced driver and receiver, an on-board crystal oscillator, a collision signal translator, and a diagnostic loopback. The features include:

- Compatibility with Ethernet I and II, IEEE 802.3 10BASE5, 10BASE2, and 10BASE-T
- 10Mb/s Manchester encoding/decoding with receive clock recovery
- No precision components requirement
- Loopback capability for diagnostics
- Squelch circuitry at the receive and collision inputs reject noise
- Connection to the transceiver (Attachment Unit Interface) cable via external pulse transformer

#### 3.2.1 ENDEC Operation

The primary function of the ENDEC unit (*Figure 3-5*) is to perform the encoding and decoding necessary for compatibility between the differential pair Manchester encoded data of the transceiver and the Non-Return-to-Zero (NRZ) serial data of the MAC unit data line. In addition to encoding and decoding the data stream, the ENDEC also supplies all of the special signals (e.g., collision detect, carrier sense, and clocks) necessary to the MAC unit. The signals provided to the MAC unit from the on-chip ENDEC are also provided as outputs to the user.

#### Manchester Encoder and Differential Output Driver:

During transmission to the network, the ENDEC unit translates the NRZ serial data from the MAC unit into differential pair Manchester encoded data. To perform this operation the NRZ bit stream from the MAC unit is passed through the Manchester encoder block of the ENDEC unit. Once the bit stream is encoded, it is transmitted out differentially to the transmit differential pair through the transmit driver.

The SONIC-T Controller is compatible with the IEEE 802.3 "full-step" standard. That is, the Transmit+ and Transmit- differential outputs are at equal voltages while they are idle at the primary of the isolation transformer at the network interface. This voltage relationship provides a zero differential voltage for operation with transformer coupled loads. (See Section 8.1 for network interfacing considerations.)

**Manchester Decoder:** During reception from the network, the differential receive data from the transceiver is converted from Manchester encoded data into NRZ serial data and a receive clock, which are sent to the receive data and clock inputs of the MAC unit. To perform this operation, the signal is passed to the PLL decoder block once it is received from the differential receiver. The PLL decodes the data and generates a data receive clock and a NRZ serial data stream to the MAC unit.

Data typically becomes valid from the decoder within 6 bit times, and the decoder detects the end of a frame when no more mid-bit transitions are detected. (See Section 8.1 for network interfacing considerations.)

**Special Signals:** In addition to performing the Manchester encoding and decoding function, the ENDEC unit provides control and clocking signals to the MAC unit. The ENDEC sends a carrier sense (CRS) signal that indicates to the MAC unit that data is present from the network on the ENDEC's receive differential pair. When the ENDEC's collision receiver detects a 10 MHz signal on the differential collision input pair, the ENDEC unit provides the MAC unit with a collision detection signal (COL). COL indicates that a collision is taking place somewhere on the network.

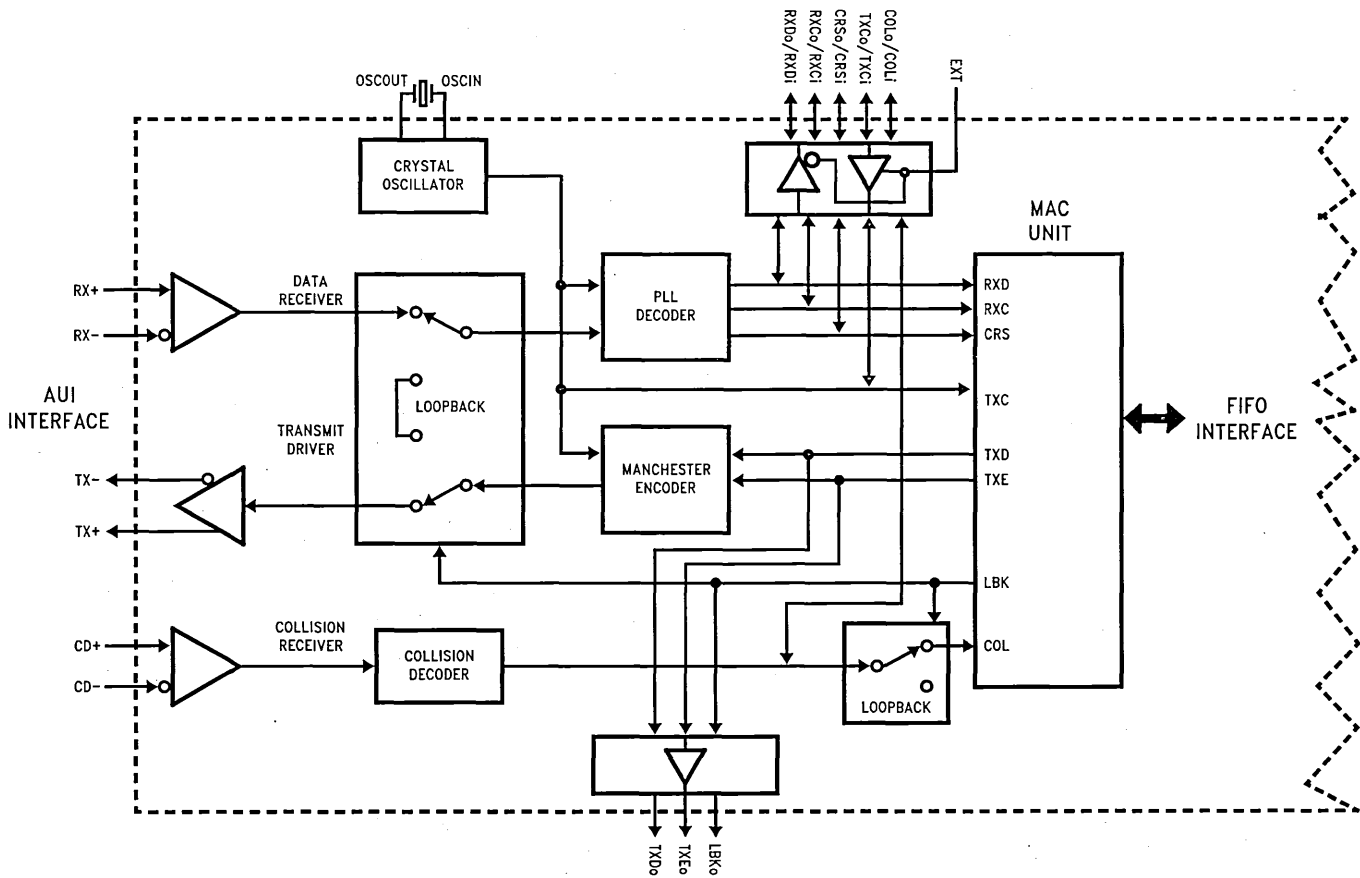


FIGURE 3-5. Block Diagram of Ethernet ENDEC

TL/F/11719-8

1-549

### 3.0 Functional Description (Continued)

The ENDEC also provides both the receive and transmit clocks to the MAC unit. The transmit clock is one half of the oscillator input and the receive clock is extracted from the input data by the PLL.

**Oscillator:** The oscillator generates the 10 MHz transmit clock signal for network timing. The oscillator is controlled by a parallel resonant crystal or by an external clock (see Section 8.1.3). The 20 MHz output of the oscillator is divided by 2 to generate the 10 MHz transmit clock (TXC) for the MAC section. The oscillator also provides an internal clock signal for the encoding and decoding circuits.

**Loopback Functions:** The SONIC-T provides three loopback modes which allow for loopback testing at the MAC, ENDEC and external transceiver level (see Section 3.7 for details). It is important to note that when the SONIC-T is transmitting, the transmitted packet will always be looped back by the external transceiver. The SONIC-T takes advantage of this to monitor the transmitted packet. See the explanation of the Receive State Machine in Section 3.3.1 for more information about monitoring transmitted packets.

#### 3.2.2 Selecting an External ENDEC

An option is provided on SONIC-T to disable the on-chip ENDEC unit and use an external ENDEC. The internal IEEE 802.3 ENDEC can be bypassed by connecting the EXT pin to  $V_{CC}$  (EXT = 1). In this mode the MAC signals are redirected out from the chip, allowing an external ENDEC to be used. See Section 2.0 for the alternate pin definitions.

#### 3.3 MEDIA ACCESS CONTROL (MAC) UNIT

The Media Access Control (MAC) unit performs the control functions for the media access of transmitting and receiving packets over Twisted Pair or AUI. During transmission, the MAC unit frames information from the transmit FIFO and supplies serialized data to the ENDEC unit. During reception, the incoming information from the ENDEC unit is deserialized, the frame checked for valid reception, and the data is transferred to the receive FIFO. Control and status registers on the SONIC-T govern the operation of the MAC unit.

##### 3.3.1 MAC Receive Section

The receive section (*Figure 3-6*) controls the MAC receive operations during reception, loopback, and transmission. During reception, the deserializer goes active after detecting the 2-bit Start of Frame Delimiter (SFD) pattern (see Section 4.1). It then frames the incoming bits into octet boundaries and transfers the data to the 32-byte receive FIFO. Concurrently the address comparator compares the Destination

Address Field to the addresses stored in the chip's Content Addressable Memory (CAM) address registers. If a match occurs, the deserializer passes the remainder of the packet to the receive FIFO. The packet is decapsulated when the carrier sense input pin (CRS) goes inactive. At the end of reception the receive section checks the following:

- Frame alignment errors
- CRC errors
- Length errors (runt packets)

The appropriate status is indicated in the Receive Control register (see Section 6.3.3). In loopback operations, the receive section operates the same as during normal reception.

During transmission, the receive section remains active to allow monitoring of the self-received packet. The Cyclic Redundancy Code (CRC) checker operates as normal, and the Source Address field is compared with the CAM address entries. Status of the CRC check and the source address comparison is indicated by the PMB bit in the Transmit Control register (see Section 6.3.4). No data is written to the receive FIFO during transmit operations.

The receive section consists of the following blocks detailed below.

**Receive State Machine (RSM):** The RSM insures the proper sequencing for normal reception and self-reception during transmission. When the network is inactive, the RSM remains in an idle state continually monitoring for network activity. If the network becomes active, the RSM allows the deserializer to write data into the receive FIFO. During this state, the following conditions may prevent the complete reception of the packet.

- FIFO Overrun—The receive FIFO has been completely filled before the SONIC-T could buffer the data to memory.
- CAM Address Mismatch—The packet is rejected because of a mismatch between the destination address of the packet and the address in the CAM.
- Memory Resource Error—There are no more resources (buffers or descriptors) available for buffering the incoming packets.
- Collision or Other Error—A collision occurred on the network or some other error, such as a CRC error, occurred (this is true if the SONIC-T has been told to reject packets on a collision, or reject packets with errors).

If these conditions do not occur, the RSM processes the packet indicating the appropriate status in the Receive Control register.

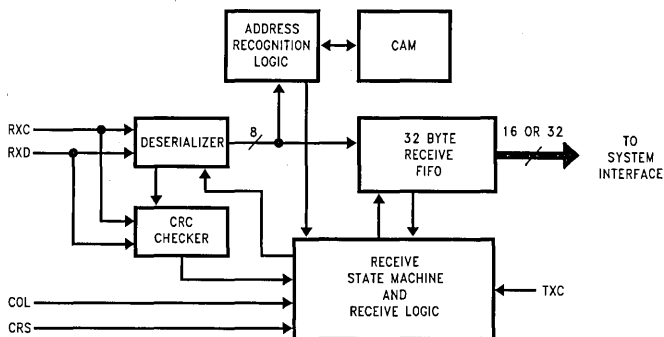


FIGURE 3-6. MAC Receiver

TL/F/11719-9

### 3.0 Functional Description (Continued)

During transmission of a packet from the SONIC-T, the transceiver will always loop the packet back to the SONIC-T. The SONIC-T will use this to monitor the packet as it is being transmitted. The CRC and source address of the looped back packet are checked with the CRC and source address that were transmitted. If they do not match, an error bit is set in the status of the transmitted packet (see Packet Monitored Bad, PMB, in the Transmit Control Register, Section 6.3.4). Data is not written to the receive FIFO during this monitoring process unless a Loopback mode has been selected (see Section 3.7).

**Receive Logic:** The receive logic contains the command, control, and status registers that govern the operations of the receive section. It generates the control signals for writing data to the receive FIFO, processes error signals obtained from the CRC checker and the deserializer, activates the "packet reject" signal to the RSM for rejecting packets, and posts the applicable status in the Receive Control register.

**Deserializer:** This section deserializes the serial input data stream and provides a byte clock for the address comparator and receive logic. It also synchronizes the CRC checker to begin operation (after SFD is detected), and checks for proper frame alignment with respect to CRS going inactive at the end of reception.

**Address Comparator:** The address comparator latches the Destination Address (during reception or loopback) or Source Address (during transmission) and determines whether the address matches one of the entries in the CAM.

**CRC Checker:** The CRC checker calculates the 4-byte Frame Check Sequence (FCS) field from the incoming data stream and compares it with the last 4-bytes of the received packet. The CRC checker is active for both normal reception and self-reception during transmission.

**Content Addressable Memory (CAM):** The CAM contains 16 user programmable entries and 1 pre-programmed Broadcast address entry for complete filtering of received packets. The CAM can be loaded with any combination of Physical and Multicast Addresses (see Section 4.2). See Section 6.1 for the procedure on loading the CAM registers.

#### 3.3.2 MAC Transmit Section

The transmit section (Figure 3-7) is responsible for reading data from the transmit FIFO and transmitting a serial data

stream onto the network in conformance with the IEEE 802.3 Carrier Sense Multiple Access with Collision Detection (CSMA/CD) standard. The Transmit Section consists of the following blocks.

**Transmit State Machine (TSM):** The TSM controls the functions of the serializer, preamble generator, and JAM generator. It determines the proper sequence of events that the transmitter follows under various network conditions. If no collision occurs, the transmitter prefixes a 62-bit preamble and 2-bit Start of Frame Delimiter (SFD) at the beginning of each packet and then sends the serialized data. At the end of the packet, an optional 4-byte CRC pattern is appended. If a collision occurs, the transmitter switches from transmitting data to sending a 4-byte Jam pattern to notify all nodes that a collision has occurred. Should the collision occur during the preamble, the transmitter waits for it to complete before jamming. After the transmission has completed, the transmitter writes status in the Transmit Control register (see Section 6.3.4).

**Protocol State Machine:** The protocol state machine assures that the SONIC-T obeys the CSMA/CD protocol. Before transmitting, this state machine monitors the carrier sense and collision signals for network activity. If any other nodes are currently transmitting, the SONIC-T defers its transmission until the network is quiet. It then transmits after its Interframe Gap Timer (9.6  $\mu$ s) has expired. The Interframe Gap time is divided into two portions. During the first 6.4  $\mu$ s, any new network activity will restart the Interframe Gap timer. Beyond this time, however, network activity is ignored and the state machine waits the remaining 3.2  $\mu$ s before transmitting. If the SONIC-T experiences a collision during a transmission, it switches from transmitting data to transmitting a 4-byte JAM pattern (4 bytes of all 1's), before ceasing to transmit. The SONIC-T then waits a random number of slot times (51.2  $\mu$ s) determined by the *Truncated Binary Exponential Backoff Algorithm* before reattempting another transmission. In this algorithm, the number of slot times to delay before the *n*th retransmission is chosen to be a random integer *r* in the range of:

$$0 \leq r \leq 2^k$$

where  $k = \min(n,10)$

If a collision occurs on the 16th transmit attempt, the SONIC-T aborts transmitting the packet and reports an "Excessive Collisions" error in the Transmit Control register.

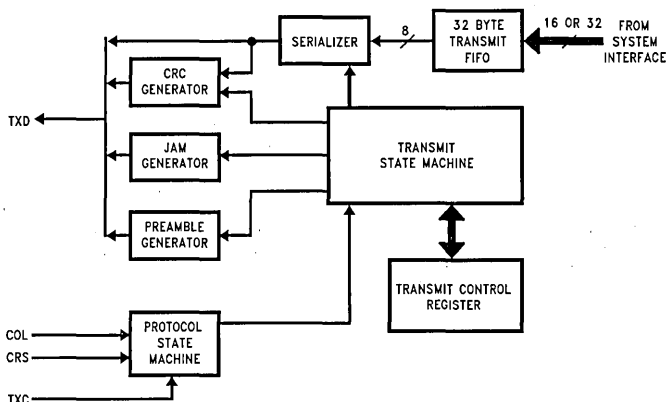


FIGURE 3-7. MAC Transmitter

TL/F/11719-10

### 3.0 Functional Description (Continued)

**Serializer:** After data has been written into the 32-byte transmit FIFO, the serializer reads byte wide data from the FIFO and sends a NRZ data stream to the Manchester encoder. The rate at which data is transmitted is determined by the transmit clock (TXC). The serialized data is transmitted after the SFD.

**Preamble Generator:** The preamble generator prefixes a 62-bit alternating "1,0" pattern and a 2-bit "1,1" SFD pattern at the beginning of each packet. This allows receiving nodes to synchronize to the incoming data. The preamble is always transmitted in its entirety even in the event of a collision. This assures that the minimum collision fragment is 96 bits (64 bits of normal preamble, and 4 bytes, or 32 bits of JAM pattern).

**CRC Generator:** The CRC generator calculates the 4-byte FCS field from the transmitted serial data stream. If enabled, the 4-byte FCS field is appended to the end of the transmitted packet (see Section 4.6).

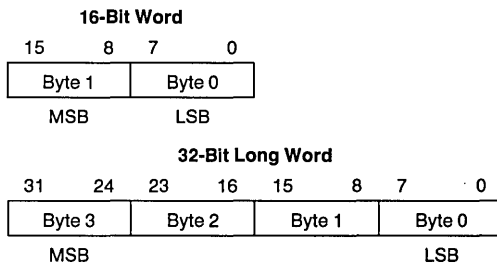
For bridging or switched ethernet applications the CRC Generator can be inhibited by setting bit 13 in the Transmit Control Register (Section 6.3.4). This feature is used when an ethernet segment has already received a packet with a CRC appended and needs to forward it another ethernet segment.

**Jam Generator:** The Jam generator produces a 4-byte pattern of all 1's to assure that all nodes on the network sense the collision. When a collision occurs, the SONIC-T stops transmitting data and enables the Jam generator. If a collision occurs during the preamble, the SONIC-T finishes transmitting the preamble before enabling the Jam generator (see Preamble Generator above).

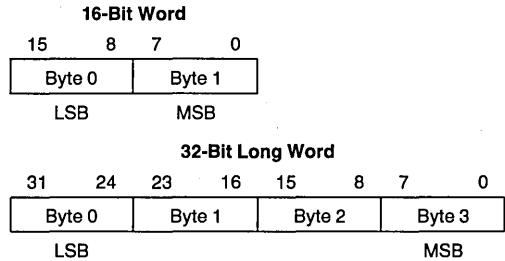
#### 3.4 DATA WIDTH AND BYTE ORDERING

The SONIC-T can be programmed to operate with either 32-bit or 16-bit wide memory. The data width is configured during initialization by programming the DW bit in the Data Configuration Register (DCR) (see Section 6.3.2). If the 16-bit data path is selected, data is driven on pins D15-D0. The SONIC-T also provides both Little Endian and Big Endian byte-ordering capability for compatibility with National/Intel or Motorola microprocessors respectively by selecting the proper level on the Bus Mode (BMODE) pin.

**Little Endian (National/Intel) Mode (BMODE = 0):** The byte orientation for received and transmitted data in the Receive Buffer Area (RBA) and Transmit Buffer Area (TBA) of system memory is as follows:



**Big Endian (Motorola) Mode (BMODE = 1):** The byte orientation for received and transmitted data in the RBA and TBA is as follows:



#### 3.5 FIFO AND CONTROL LOGIC

The SONIC-T incorporates two independent 32-byte FIFOs for transferring data to/from the system interface and from/to the network. The FIFOs, providing temporary storage of data, free the host system from the real-time demands on the network.

The way in which the FIFOs are emptied and filled is controlled by the FIFO threshold values and the Block Mode Select bits (BMS) (see Section 6.3.2). The threshold values determine how full or empty the FIFOs are allowed to be before the SONIC-T will request access of the bus to get more data from memory or buffer more data to memory. When Block Mode is enabled, the number of bytes transferred is determined by the threshold value. For example, if the threshold for the receive FIFO is 4 words, then the SONIC-T will always transfer 4 words from the receive FIFO to memory. If Empty/Fill mode is enabled, however, the number of bytes transferred is the number required to fill the transmit FIFO or empty the receive FIFO. The manner in which the threshold affects reception and transmission of packets is discussed below in Sections 3.5.1 and 3.5.2.

##### 3.5.1 Receive FIFO

To accommodate the different transfer rates, the receive FIFO (*Figure 3-8*) serves as a buffer between the 8-bit network (deserializer) interface and the 16/32-bit system interface. The FIFO is arranged as a 4-byte wide by 8 deep memory array (8-long words, or 32 bytes) controlled by three sections of logic. During reception, the Byte Ordering logic directs the byte stream from the deserializer into the FIFO using one of four write pointers. Depending on the selected byte-ordering mode, data is written either least significant byte first or most significant byte first to accommodate little or big endian byte-ordering formats respectively. As data enters the FIFO, the Threshold Logic monitors the number of bytes written in from the deserializer. The programmable threshold (RFT1,0 in the Data Configuration Register, see Section 6.3.2) determines the number of words (or long words) written into the FIFO from the MAC unit before a direct memory access (DMA) request for system memory occurs. When the threshold is reached, the

### 3.0 Functional Description (Continued)

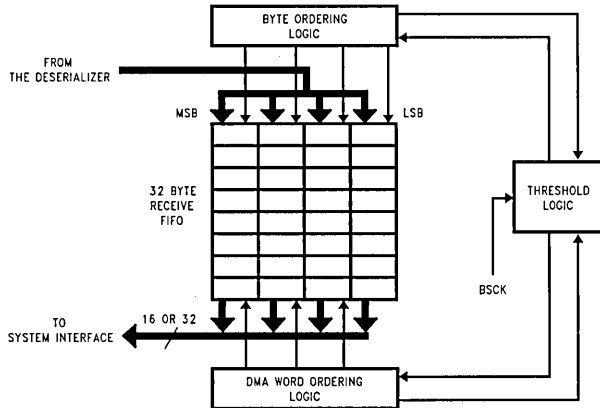


FIGURE 3-8. Receive FIFO

TL/F/11719-11

Threshold Logic enables the Buffer Management Engine to read a programmed number of 16- or 32-bit words (depending upon the selected data width) from the FIFO and transfer them to the system interface (the system memory) using DMA. The threshold is reached when the number of bytes in the receive FIFO is greater than the value of the threshold. For example, if the threshold is 4 words (8 bytes), then the Threshold Logic will not cause the Buffer Management Engine to write to memory until there are more than 8 bytes in the FIFO.

The Buffer Management Engine reads either the upper or lower half (16 bits) of the FIFO in 16-bit mode or reads the complete long word (32 bits) in 32-bit mode. If, after the transfer is complete, the number of bytes in the FIFO is less than the threshold, then the SONIC-T is done. This is always the case when the SONIC-T is in Empty/Fill Mode. If, however, for some reason (e.g., latency on the bus) the number of bytes in the FIFO is still greater than the threshold value, the Threshold Logic will cause the Buffer Management Engine to do a DMA request to write to memory again. This latter case is usually only possible when the SONIC-T is in Block Mode.

When in Block Mode, each time the SONIC-T requests the bus, only a number of bytes equal to the threshold value will be transferred. The Threshold Logic continues to monitor the number of bytes written in from the deserializer and enables the Buffer Management Engine every time the threshold has been reached. This process continues until the end of the packet.

Once the end of the packet has been reached, the serializer will fill out the last word (16-bit mode) or long word (32-bit mode) if the last byte did not end on a word or long word boundary respectively. The fill byte will be 0FFh. Immediately after the last byte (or fill byte) in the FIFO, the received packets status will be written into the FIFO. The entire packet, including any fill bytes and the received packet status will be buffered to memory. When a packet is buffered to memory by the Buffer Management Engine, it is always taken from the FIFO in words or long words and buffered to memory on word (16-bit mode) or long word (32-bit mode) boundaries. Data from a packet cannot be buffered on odd byte boundaries for 16-bit mode, and odd word boundaries for 32-bit mode (see Section 5.3). For more information on the receive packet buffering process, see Section 5.4.

#### 3.5.2 Transmit FIFO

Similar to the Receive FIFO, the Transmit FIFO (Figure 3-9) serves as a buffer between the 16/32-bit system interface and the network (serializer) interface. The Transmit FIFO is also arranged as a 4 byte by 8 deep memory array (8 long words or 32 bytes) controlled by three sections of logic. Before transmission can begin, the Buffer Management Engine fetches a programmed number of 16- or 32-bit words from memory and transfers them to the FIFO. The Buffer Management Engine writes either the upper or lower half (16 bits) into the FIFO for 16-bit mode or writes the complete long word (32 bits) during 32-bit mode.

The Threshold Logic monitors the number of bytes as they are written into the FIFO. When the threshold has been reached, the Transmit Byte Ordering state machine begins reading bytes from the FIFO to produce a continuous byte stream for the serializer. The threshold is met when the number of bytes in the FIFO is greater than the value of the threshold. For example, if the transmit threshold is 4 words (8 bytes), the Transmit Byte Ordering state machine will not begin reading bytes from the FIFO until there are 9 or more bytes in the buffer. The Buffer Management Engine continues replenishing the FIFO until the end of the packet. It does this by making multiple DMA requests to the system interface. Whenever the number of bytes in the FIFO is equal to or less than the threshold value, the Buffer Management Engine will do a DMA request. If Block Mode is set, then after each request has been granted by the system, the Buffer Management Engine will transfer a number of bytes equal to the threshold value into the FIFO. If Empty/Fill Mode is set, the FIFO will be completely filled in one DMA request.

Since data may be organized in big or little endian byte ordering format, the Transmit Byte Ordering state machine uses one of four read pointers to locate the proper byte within the 4 byte wide FIFO. It also determines the valid number of bytes in the FIFO. For packets which begin or end at odd bytes in the FIFO, the Buffer Management Engine writes extraneous bytes into the FIFO. The Transmit Byte Ordering state machine detects these bytes and only transfers the valid bytes to the serializer. The Buffer Management Engine can read data from memory on any byte boundary (see Section 5.3). See Section 5.5 for more information on transmit buffering.

### 3.0 Functional Description (Continued)

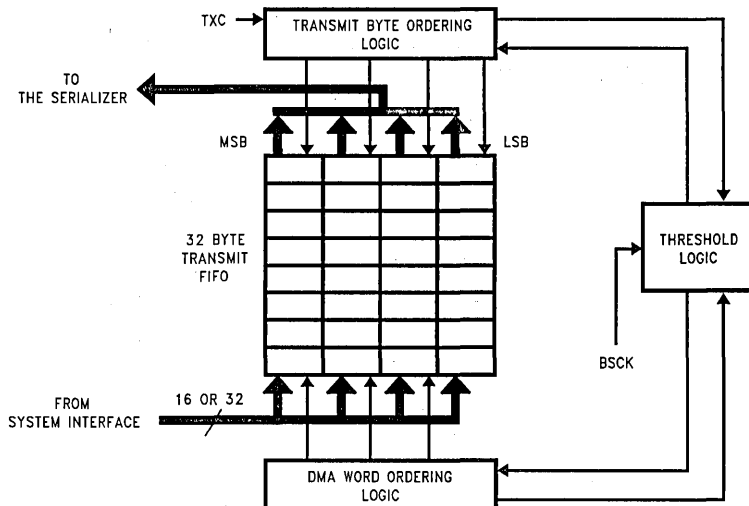


FIGURE 3-9. Transmit FIFO

TL/F/11719-12

#### 3.6 STATUS AND CONFIGURATION REGISTERS

The SONIC-T contains a set of status/control registers for conveying status and control information to/from the host system. The SONIC-T uses these registers for loading commands generated from the system, indicating transmit and receive status, buffering data to/from memory, and providing interrupt control. Each register is 16 bits in length. See Section 6.0 for a description of the registers.

#### 3.7 BUS INTERFACE

The system interface (Figure 3-10) consists of the pins necessary for interfacing to a variety of buses. It includes the I/O drivers for the data and address lines, bus access control for standard microprocessors, ready logic for synchronous or asynchronous systems, slave access control, interrupt control, and shared-memory access control. The functional signal groups are shown in Figure 3-10. See Section 7.0 for a complete description of the SONIC-T bus interface.

#### 3.8 LOOPBACK AND DIAGNOSTICS

The SONIC-T provides three loopback modes for self-testing from the controller interface to the transceiver interface. The loopback function is provided to allow self-testing of the chip's internal transmit and receive operations. During loopback, transmitted packets are routed back to the receive section of the SONIC-T where they are filtered by the address recognition logic and buffered to memory if accepted. Transmit and receive status and interrupts remain active during loopback. This means that when using loopback, it is as if the packet was transmitted and received by two separate chips that are connected to the same bus and memory.

**MAC Loopback:** Transmitted data is looped back at the MAC. Data is not sent from the MAC to either the internal ENDEC or an external ENDEC (the external ENDEC interface pins will not be driven), hence, data is not transmitted from the chip. Even though the ENDEC is not used in MAC loopback, the ENDEC clock (an oscillator or crystal for the internal ENDEC or TXC for an external ENDEC) must be driven. Network activity, such as a collision, does not affect

MAC loopback, and the CSMA/CD MAC protocol is not completely followed.

**ENDEC Loopback:** Transmitted data is looped back at the ENDEC. If the internal ENDEC is used, data is switched from the transmit section of the ENDEC to the receive section (Figure 3-5). Data is not transmitted from the chip and the collision lines,  $CD_{\pm}$ , are ignored, hence, network activity does not affect ENDEC loopback. The LBK signal from the MAC tells the internal ENDEC to go into loopback mode. If an external ENDEC is used, it should operate in loopback mode when the LBK signal is asserted. CSMA/CD MAC protocol is followed even though data is not transmitted from the chip.

**Transceiver Loopback:** Transmitted data is looped back at the external transceiver (which is always the case regardless of the SONIC-T's loopback mode). CSMA/CD MAC protocol is followed since data will be transmitted from the chip. This means that transceiver loopback is affected by network activity. The basic difference between Transceiver Loopback Mode and the other loopback modes is that the SONIC-T loads the receive FIFO and buffers the packet to memory. In normal operations, the SONIC-T only monitors the packet that is looped back by the transceiver, but does not fill the receive FIFO and buffer the packet.

##### 3.8.1 Loopback Procedure

The following procedure describes the loopback operation.

1. Initialize the Transmit and Receive Area as described in Sections 5.4 and 5.5.
2. Load one of the CAM address registers (see Section 6.1), with the Destination Address of the packet if you are verifying the SONIC-T's address recognition capability.
3. Load one of the CAM address registers with the Source Address of the packet if it is different than the Destination Address to avoid getting a Packet Monitored Bad (PMB) error in the Transmit Status (see Section 6.3.4).



### 3.0 Functional Description (Continued)

4. Program the Receive Control register with the desired receive filter and the loopback mode (LB1, LB0). In case of transceiver loopback, besides setting LB1 and LB0 to 1, the XWRAP bit in the DCR2 must also set to 1.

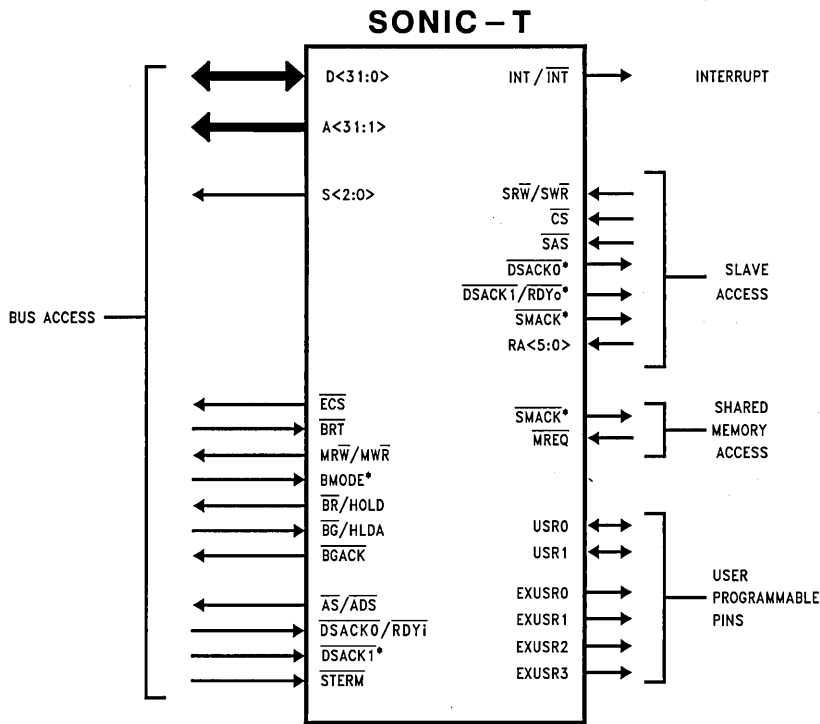
5. Issue the transmit command (TXP) and enable the receiver (RXEN) in the Command register.

The SONIC-T completes the loopback operation after the packet has been completely received (or rejected if there is an address mismatch). The Transmit Control and Receive Control registers treat the loopback packet as it would in normal operation and indicate status accordingly. Interrupts are also generated if enabled in the Interrupt Mask register.

**Note:** For MAC Loopback, only one packet may be queued up for proper operation. This restriction occurs because the transmit MAC section, which does not generate an Interframe Gap (IFG) time between transmitted packets, does not allow the receive MAC section to update receive status. There are no restrictions for the other loopback modes.

### 3.9 NETWORK MANAGEMENT FUNCTIONS

The SONIC-T fully supports the Layer Management IEEE 802.3 standard to allow a node to monitor the overall performance of the network. These statistics are available on a per packet basis at the end of reception or transmission. In addition, the SONIC-T provides three tally counters to tabulate CRC errors, Frame Alignment errors, and missed packets. Table 3-1 shows the statistics indicated by the SONIC-T.



TL/F/11719-13

\*Note: DSACK0,1 are used for both Bus and Slave Access Control and are bidirectional. SMACK is used for both Slave access and shared memory access. The BMODE pin selects between National/Intel or Motorola type busses.

FIGURE 3-10. SONIC-T Bus Interface Signals

### 3.0 Functional Description (Continued)

TABLE 3-1. Network Management Statistics

Statistic	Register Used	Bits Used
Frames Transmitted OK	TCR (Note)	PTX
Single Collision Frames	(Note)	NC0-NC4
Multiple Collision Frames	(Note)	NC0-NC4
Collision Frames	(Note)	NC0-NC4
Frames with Deferred Transmissions	TCR (Note)	DEF
Late Collisions	TCR (Note)	OWC
Excessive Collisions	TCR (Note)	EXC
Excessive Deferral	TCR (Note)	EXD
Internal MAC Transmit Error	TCR (Note)	BCM,FU
Frames Received OK	RCR (Note)	PRX
Multicast Frames Received OK	RCR (Note)	MC
Broadcast Frames Received OK	RCR (Note)	BC
Frame Check Sequence Errors	CRCT RCR	All CRC
Alignment Errors	FAET RCR	All FAE
Frame Lost due to Internal MAC Receive Error	MPT ISR	All RFO

Note: The number of collisions and the contents of the Transmit Control register are posted in the Txpkt.status field (see Section 5.5.1.2). The contents of the Receive Control register are posted in the RXPkt.status field (see Section 5.4.3).

### 4.0 Transmit/Receive IEEE 802.3 Frame Format

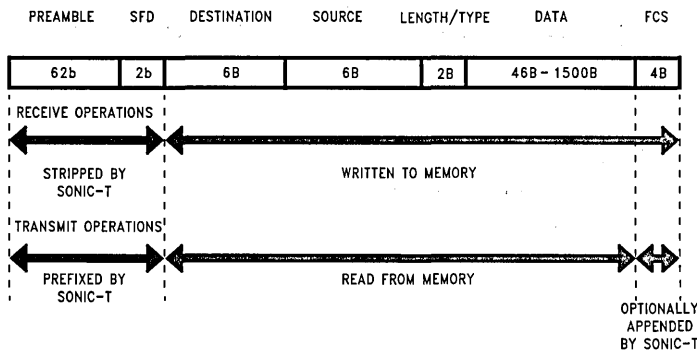
A standard IEEE 802.3 packet (Figure 4-1) consists of the following fields: preamble, Start of Frame Delimiter (SFD), destination address, source address, length, data and Frame Check Sequence (FCS). The typical format is shown in Figure 4-1. The packets are Manchester encoded and decoded by the ENDEC unit and transferred serially to/from the MAC unit using NRZ data with a clock. All fields are of fixed length except for the data field. The SONIC-T generates and appends the preamble, SFD and FCS field during transmission. The Preamble and SFD fields are stripped during reception. (The CRC is passed through to buffer memory during reception.)

#### 4.1 PREAMBLE AND START OF FRAME DELIMITER (SFD)

The Manchester encoded alternating 1,0 preamble field is used by the ENDEC to acquire bit synchronization with an incoming packet. When transmitted, each packet contains 62 bits of an alternating 1,0 preamble. Some of this preamble may be lost as the packet travels through the network. Byte alignment is performed when the Start of Frame Delimiter (SFD) pattern, consisting of two consecutive 1's, is detected.

#### 4.2 DESTINATION ADDRESS

The destination address indicates the destination of the packet on the network and is used to filter unwanted pack-



Note: B = bytes  
b = bits

FIGURE 4-1. IEEE 802.3 Packet Structure

TL/F/11719-14

## 4.0 Transmit/Receive IEEE 802.3 Frame Format (Continued)

ets from reaching a node. There are three types of address formats supported by the SONIC-T: Physical, Multicast, and Broadcast.

**Physical Address:** The physical address is a unique address that corresponds only to a single node. All physical addresses have the LSB of the first byte of the address set to "0". These addresses are compared to the internally stored CAM (Content Addressable Memory) address entries. All bits in the destination address must match an entry in the CAM in order for the SONIC-T to accept the packet.

**Multicast Address:** Multicast addresses, which have the LSB of the first byte of the address set to "1", are treated similarly as physical addresses, i.e., they must match an entry in the CAM. This allows perfect filtering of Multicast packets and eliminates the need for a hashing algorithm for mapping Multicast packets.

**Broadcast Address:** If the address consists of all 1's, it is a Broadcast address, indicating that the packet is intended for all nodes.

The SONIC-T also provides a promiscuous mode which allows reception of all physical address packets. Physical, Multicast, Broadcast, and promiscuous address modes can be selected via the Receive Control register.

### 4.3 SOURCE ADDRESS

The source address is the physical address of the sending node. Source addresses cannot be multicast or broadcast addresses. This field must be passed to the SONIC-T's transmit buffer from the system software. During transmission, the SONIC-T compares the Source address with its internal CAM address entries before monitoring the CRC of the self-received packet. If the source address of the packet transmitted does not match a value in the CAM, the packet monitored bad flag (PMB) will be set in the transmit status field of the transmit descriptor (see Sections 5.5.1.2 and 6.3.4). The SONIC-T does not provide Source Address insertion. However, a transmit descriptor fragment, containing only the Source Address, may be created for each packet. (See Section 5.5.1.)

### 4.4 LENGTH/TYPE FIELD

For IEEE 802.3 type packets, this field indicates the number of bytes that are contained in the data field of the packet. For Ethernet I and II networks, this field indicates the type of packet. The SONIC-T does not operate on this field.

### 4.5 DATA FIELD

The data field has a variable octet length ranging from 46 to 1500 bytes as defined by the Ethernet specification. Messages longer than 1500 bytes need to be broken into multiple packets for IEEE 802.3 networks. Data fields shorter than 46 bytes require appending a pad to bring the complete frame length to 64 bytes. If the data field is padded, the number of valid bytes are indicated in the length field. The SONIC-T does not append pad bytes for short packets during transmission, nor check for oversize packets during reception. However, the user's driver software can easily append the pad by lengthening the TXpkt.pkt\_size field and TXpkt.frag\_size field(s) to at least 64 bytes (see Section 5.5.1). Although the Ethernet specification defines the maximum number of bytes in the data field, the SONIC-T can transmit and receive packets up to 64k bytes.

### 4.6 FCS FIELD

The Frame Check Sequence (FCS) is a 32-bit CRC field calculated and appended to a packet during transmission to allow detection of error-free packets. During reception, an

error-free packet results in a specific pattern in the CRC generator. The AUTODIN II ( $X_{32} + X_{26} + X_{23} + X_{22} + X_{16} + X_{12} + X_{11} + X_{10} + X_8 + X_7 + X_5 + X_4 + X_2 + X_1 + 1$ ) polynomial is used for the CRC calculations. The SONIC-T may optionally append the CRC sequence during transmission, and checks the CRC both during normal reception and self-reception during a transmission (see Section 3.3.1).

### 4.7 MAC (MEDIA ACCESS CONTROL) CONFORMANCE

The SONIC-T is designed to be compliant to the IEEE 802.3 MAC Conformance specification. The SONIC-T implements most MAC functions in silicon and provides hooks for the user software to handle the remaining functions. The MAC Conformance specifications are summarized in Table 4-1.

TABLE 4-1. MAC Conformance Specifications

Conformance Test Name	Support By		
	SONIC-T	User Driver Software	Notes
Minimum Frame Size	X		
Maximum Frame Size	X	X	1
Address Generation	X	X	2
Address Recognition	X		
Pad Length Generation	X	X	3
Start Of Frame Delimiter	X		
Length Field	X		
Preamble Generation	X		
Order of Bit Transmission	X		
Inconsistent Frame Length	X	X	1
Non-Integral Octet Count	X		
Incorrect Frame Check Sequence	X		
Frame Assembly	X		
FCS Generation and Insertion	X		
Carrier Deference	X		
Interframe Spacing	X		
Collision Detection	X		
Collision Handling	X		
Collision Backoff and Retransmission	X		
FCS Validation	X		
Frame Disassembly	X		
Back-to-Back Frames	X		
Flow Control	X		
Attempt Limit	X		
Jam Size (after SFD)	X		
Jam Size (in Preamble)	X		

**Note 1:** The SONIC-T provides the byte count of the entire packet in the RXpkt.byte\_count (see Section 5.4.3). The user's driver software may perform further filtering of the packet based upon the byte count.

**Note 2:** The SONIC-T does not provide Source Address insertion; however, a transmit descriptor fragment, containing only the Source Address, may be created for each packet. (See Section 5.5.1.)

**Note 3:** The SONIC-T does not provide Pad generation; however, the user's driver software can easily append the Pad by lengthening the TXpkt.pkt\_size field and TXpkt.frag\_size field(s) to at least 64 bytes. (See Section 5.5.1.)

## 5.0 Buffer Management

### 5.1 BUFFER MANAGEMENT OVERVIEW

The SONIC-T's buffer management scheme is based on separate buffers and descriptors (*Figures 5-3 and 5-12*). Packets that are received or transmitted are placed in buffers called the Receive Buffer Area (RBA) and the Transmit Buffer Area (TBA). The system keeps track of packets in these buffers using the information in the Receive Descriptor Area (RDA) and the Transmit Descriptor Area (TDA). A single (TDA) points to a single TBA, but multiple RDAs can point to a single RBA (one RDA per packet in the buffer). The Receive Resource Area (RRA), which is another form of descriptor, is used to keep track of the actual buffer.

When packets are transmitted, the system sets up the packets in one or more TBAs with a TDA pointing to each TBA. There can only be one packet per TBA/TDA pair. A single TBA, however, may be made up of several fragments of data dispersed in memory. There is one TDA pointing to each TBA which specifies information about the buffer's size, location in memory, number of fragments and status after transmission. The TDAs are linked together in a linked list. The system causes the SONIC-T to transmit the packets by passing the first TDA to the SONIC-T and issuing the transmit command.

Before a packet can be received, an RDA and RBA must be set up by the system. RDAs are made up as a linked list similar to TDAs. An RDA is not linked to a particular RBA, though. Instead, an RDA is linked specifically to a packet after it has been buffered into an RBA. More than one packet can be buffered into the same RBA, but each packet gets its own RDA. A received packet can not be scattered into fragments. The system only needs to tell the SONIC-T where the first RDA and where the RDAs are. Since an RDA never specifically points to an RBA, the RRA is used to keep track of the RBAs. The RRA is a circular queue of pointers and buffer sizes (not a linked list). When the SONIC-T receives a packet, it is buffered into an RBA and a unique corresponding RDA is written to so that it points to and describes the new packet. If the RBA does not have enough space to buffer the next packet, a new RBA is obtained from the RRA.

### 5.2 DESCRIPTOR AREAS

Descriptors are the basis of the buffer management scheme used by the SONIC-T. An RDA points to a received packet within an RBA, an RRA points to an RBA and a TDA points to a TBA which contains a packet to be transmitted. The conventions and registers used to describe these descriptors are discussed in the next three sections.

#### 5.2.1 Naming Convention for Descriptors

The fields which make up the descriptors are named in a consistent manner to assist in remembering the usage of each descriptor. Each descriptor name consists of three components in the following format.

[RX/TX][descriptor name].[field]

The first two capital letters indicate whether the descriptor is used for transmission (TX) or reception (RX), and is then followed by the descriptor name having one of two names.

rsrc = Resource descriptor  
pkt = Packet descriptor

The last component consists of a field name to distinguish it from the other fields of a descriptor. The field name is separated from the descriptor name by a period ("."). An example of a descriptor is shown in *Figure 5-1*.

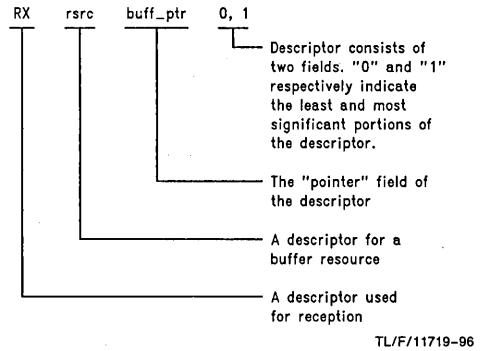


FIGURE 5-1. Receive Buffer Descriptor Example

#### 5.2.2 Abbreviations

Abbreviations are used to describe the SONIC-T registers and data structures in memory. The "0" and "1" in the abbreviations indicate the least and most significant portions of the registers or descriptors. Table 5-1 lists the naming convention abbreviations for descriptors.

#### 5.2.3 Buffer Management Base Addresses

The SONIC-T uses three areas in memory to store descriptor information: the Transmit Descriptor Area (TDA), the Receive Descriptor Area (RDA), and the Receive Resource Area (RRA). The SONIC-T accesses these areas by concatenating a 16-bit base address register with a 16-bit offset register. The base address register supplies a fixed upper 16 bits of address and the offset registers provide the lower 16 bits of address. The base address registers are the Upper Transmit Descriptor Address (UTDA), Upper Receive Descriptor Address (URDA), and the Upper Receive Resource Address (URRA) registers. The corresponding offset registers are shown below.

Upper Address Registers	Offset Registers
URRA	RSA, REA, RWP, RRP
URDA	CRDA
UTDA	CTDA

Table 5-1 defines the register mnemonics.

*Figure 5-2* shows an example of the Transmit Descriptor Area and the Receive Descriptor Area being located by the UTDA and URDA registers. The descriptor areas, RDA, TDA, and RRA are allowed to have the same base address, i.e., URRA=URDA=UTDA. Care, however, must be taken to prevent these areas from overwriting each other.

## 5.0 Buffer Management (Continued)

TABLE 5-1. Descriptor Abbreviations

TRANSMIT AND RECEIVE AREAS	
RRA	Receive Resource Area
RDA	Receive Descriptor Area
RBA	Receive Buffer Area
TDA	Transmit Descriptor Area
TBA	Transmit Buffer Area
BUFFER MANAGEMENT REGISTERS	
RSA	Resource Start Area Register
REA	Resource End Area Register
RRP	Resource Read Pointer Register
RWP	Resource Write Pointer Register
CRDA	Current Receive Descriptor Address Register
CRBA0,1	Current Receive Buffer Address Register
TCBA0,1	Temporary Current Buffer Address Register
RBWC0,1	Remaining Buffer Word Count Register
TRBWC0,1	Temporary Remaining Buffer Word Count Register
EOBC	End of Buffer Count Register
TPS	Transmit Packet Size Register
TSA0,1	Transmit Start Address Register
CTDA	Current Transmit Descriptor Address Register

BUFFER MANAGEMENT REGISTERS (Continued)	
TFC	Transmit Fragment Count Register
TFS	Transmit Fragment Size Register
UTDA	Upper Transmit Descriptor Address Register
URRA	Upper Receive Resource Address Register
URDA	Upper Receive Descriptor Address Register
TRANSMIT AND RECEIVE DESCRIPTORS	
RXrsrc.buff_ptr0,1	Buffer Pointer Field in the RRA
RXrsrc.buff_wc0,1	Buffer Word Count Fields in the RRA
RXpkt.status	Receive Status Field in the RDA
RXpkt.byte_count	Packet Byte Count Field in the RDA
RXpkt.buff_ptr0,1	Buffer Pointer Fields in the RDA
RXpkt.link	Receive Descriptor Link Field in RDA
RXpkt.in_use	"In Use" Field in RDA
TXpkt.frag_count	Fragment Count Field in TDA
TXpkt.pkt_size	Packet Size Field in TDA
TXpkt.pkt_ptr0,1	Packet Pointer Fields in TDA
TXpkt.frag_size	Fragment Size Field in TDA
TXpkt.link	Transmit Descriptor Link Field in TDA

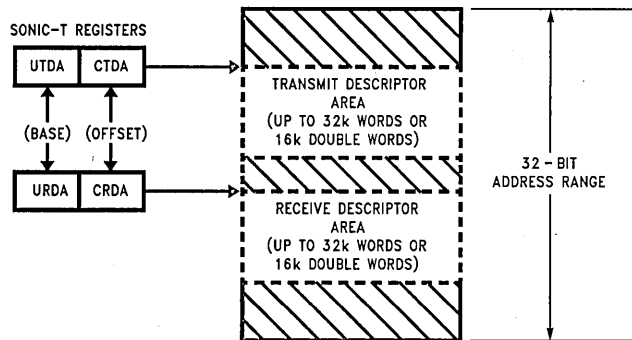


FIGURE 5-2. Transmit and Receive Descriptor Area Pointers

TL/F/11719-15

## 5.0 Buffer Management (Continued)

### 5.3 DESCRIPTOR DATA ALIGNMENT

All fields used by descriptors (RXpkt.xxx, RXsrc.xxx, and TXpkt.xxx) are word quantities (16-bit) and must be aligned to word boundaries (A0=0) for 16-bit memory and to long word boundaries (A1,A0=0,0) for 32-bit memory. The Receive Buffer Area (RBA) must also be aligned to a word boundary in 16-bit mode and a long word boundary in 32-bit mode. The fragments in the Transmit Buffer Area (TBA), however, may be aligned on any arbitrary byte boundary.

### 5.4 RECEIVE BUFFER MANAGEMENT

The Receive Buffer Management operates on three areas in memory into which data, status, and control information are written during reception (*Figure 5-3*). These three areas must be initialized (Section 5.4.4) before enabling the receiver (setting the RXEN bit in the Command Register). The Receive Resource Area (RRA) contains descriptors that locate Receive Buffer Areas in system memory. These descriptors are denoted by R1, R2, etc. in *Figure 5-3*. Packets (denoted by P1, P2, etc.) can then be buffered into the corresponding RBAs. Depending on the size of each buffer area and the size of the packet(s), multiple or single packets are buffered into each RBA. The Receive Descriptor Area (RDA) contains status and control information for each packet (D1, D2, etc. in *Figure 5-3*) corresponding to each received packet (D1 goes with P1, D2 with P2, etc.).

When a packet arrives, the address recognition logic checks the address for a Physical, Multicast, or Broadcast match and if the packet is accepted, the SONIC-T buffers the packet contiguously into the selected Receive Buffer Area (RBA). Because of the previous end-of-packet processing, the SONIC-T assures that the complete packet is written into a single contiguous block. When the packet ends, the SONIC-T writes the receive status, byte count, and location of the packet into the Receive Descriptor Area (RDA). The SONIC-T then updates its pointers to locate the next available descriptor and checks the remaining words available in the RBA. If sufficient space remains, the SONIC-T buffers the next packet immediately after the previous packet. If the current buffer is out of space the SONIC-T fetches a Resource Descriptor from the Receive Resource Area (RRA) acquiring an additional buffer that has been previously allocated by the system.

### 5.4.1 Receive Resource Area (RRA)

As buffer memory is consumed by the SONIC-T for storing data, the Receive Resource Area (RRA) provides a mechanism that allows the system to allocate additional buffer space for the SONIC-T. The system loads this area with Resource Descriptors that the SONIC-T, in turn, reads as its current buffer space is used up. Each Resource Descriptor consists of a 32-bit buffer pointer locating the starting point of the RBA and a 32-bit word count that indicates the size of the buffer in words (2 bytes per word). The buffer pointer and word count are contiguously located using the format shown in *Figure 5-4* with each component composed of 16-bit fields. The SONIC-T stores this information internally and concatenates the corresponding fields to create 32-bit long words for the buffer pointer and word count. Note that in 32-bit mode the upper word (D<31:16>) is not used by the SONIC-T. This area may be used for other purposes since the SONIC-T never writes into the RRA.

The SONIC-T organizes the RRA as a circular queue for efficient processing of descriptors. Four registers define the RRA. The first two, the Resource Start Area (RSA) and the Resource End Area (REA) registers, determine the starting and ending locations of the RRA, and the other two registers update the RRA. The system adds descriptors at the address specified by the Resource Write Pointer (RWP), and the SONIC-T reads the next descriptor designated by the Resource Read Pointer (RRP). The RRP is advanced 4 words in 16-bit mode (4 long words in 32-bit mode) after the SONIC-T finishes reading the RRA and automatically wraps around to the beginning of the RRA once the end has been reached. When a descriptor in the RRA is read, the RXsrc.buf\_pt0,1 is loaded into the CRBA0,1 registers and the RXsrc.buf\_wc0,1 is loaded into the RBWC0,1 registers.

The alignment of the RRA is confined to either word or long word boundaries, depending upon the data width mode. In 16-bit mode, the RRA must be aligned to a word boundary (A0 is always zero) and in 32-bit mode, the RRA is aligned to a long word boundary (A0 and A1 are always zero).

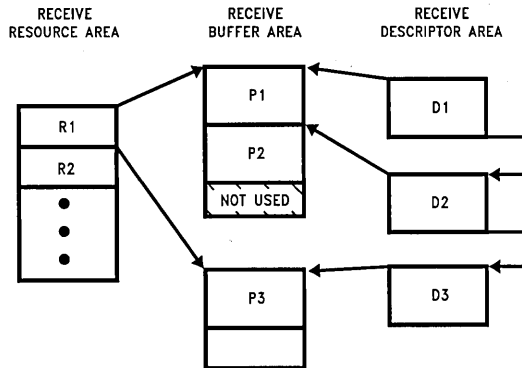


FIGURE 5-3. Overview of Receive Buffer Management

TL/F/11719-16

## 5.0 Buffer Management (Continued)

### 5.4.2 Receive Buffer Area (RBA)

The SONIC-T stores the actual data of a received packet in the RBA. The RBAs are designated by the Resource Descriptors in the RRA as described above. The `RXsrc.buf_wc0,1` fields of the RRA indicate the length of the RBA. When the SONIC-T gets an RBA from the RRA, the `RXsrc.buf_wc0,1` values are loaded into the Remaining Buffer Word Count registers (`RBWC0,1`). These registers keep track of how much space (in words) is left in the buffer. When a packet is buffered in a RBA, it is buffered contiguously (the SONIC-T will not scatter a packet into multiple buffers or fragments). Therefore, if there is not enough space left in a RBA after buffering a packet to buffer at least one more maximum sized packet (the maximum legal sized packet expected to be received from the network), a new buffer must be acquired. The End of Buffer Count (EOBC) register is used to tell the SONIC-T the maximum packet size that the SONIC-T will need to buffer.

#### 5.4.2.1 End of Buffer Count (EOBC)

The EOBC is a boundary in the RBA based from the bottom of the buffer. The value written into the EOBC is the maximum expected size (in words) of the network packet that the SONIC-T will have to buffer. This word count creates a line in the RBA that, when crossed, causes the SONIC-T to fetch a new RBA resource from the RRA.

**Note:** The EOBC is a word count, not a byte count. Also, the value programmed into EOBC must be a double word (32-bit) quantity when the SONIC-T is in 32-bit mode (e.g., in 32-bit mode, EOBC should be set to 760 words, not 759 words even though the maximum size of an IEEE 802.3 packet is 759 words for double word boundary alignment).

#### 5.4.2.2 Buffering the Last Packet in an RBA

At the start of reception, the SONIC-T stores the packet beginning at the Current Receive Buffer Address (`CRBA0,1`) and continues until the reception is complete. Concurrent with reception, the SONIC-T decrements the Remaining Buffer Word Count (`RBWC0,1`) by one in 16-bit mode or by two in 32-bit mode. At the end of reception, if the packet has crossed the EOBC boundary, the SONIC-T knows that the next packet might not fit in the RBA. This check is done by comparing the `RBWC0,1` registers with the EOBC. If `RBWC0,1` is less than the EOBC (the last packet buffered has crossed the EOBC boundary), the SONIC-T fetches the next resource descriptor in the RRA. If `RBWC0,1` is greater than or equal to the EOBC (the EOBC boundary has not been crossed) the next packet reception continues at the present location pointed to by `CRBA0,1` in the same RBA. *Figure 5-5* illustrates the SONIC-T's actions for (1)  $RBWC0,1 \geq EOBC$  and (2)  $RBWC0,1 < EOBC$ . See Section 5.4.4.4 for specific information about setting the EOBC.

**Note:** It is important that the EOBC boundary be "crossed." In other words, case #1 in *Figure 5-5* must exist before case #2 exists. If case #2 occurs without case #1 having occurred first, the test for  $RBWC0,1 < EOBC$  will not work properly and the SONIC-T will not fetch a new buffer. The result of this will be a buffer overflow (RBAE in the Interrupt Status Register, Section 6.3.6).

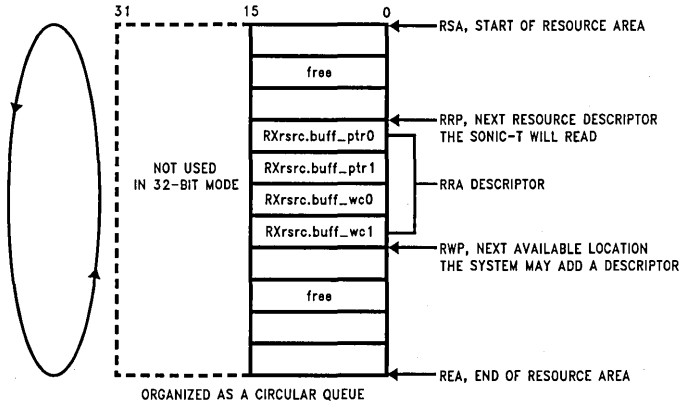
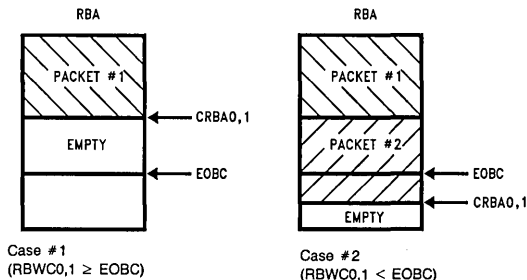


FIGURE 5-4. Receive Resource Area Format

TL/F/11719-17



Case #1: SONIC-T buffers next packet in same RBA.  
Case #2: SONIC-T detects an exhausted RBA and will buffer the next packet in another RBA.

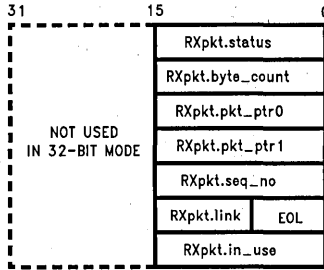
FIGURE 5-5. Receive Buffer Area

TL/F/11719-18

## 5.0 Buffer Management (Continued)

### 5.4.3 Receive Descriptor Area (RDA)

After the SONIC-T buffers a packet to memory, it writes 6 words of status and control information into the RDA reads the link field to the next Receive Descriptor, and writes to the in-use field of the current descriptor. In 32-bit mode the upper word, D<31:16>, is not used. This unused area in memory should not be used for other purposes, since the SONIC-T may still write into these locations. Each Receive Descriptor consists of the following sections: (Figure 5-6).



TL/F/11719-19

FIGURE 5-6. Receive Descriptor Format

**receive status:** indicates status of the received packet. The SONIC-T writes the Receive Control register values into this field. Figure 5-7 shows the receive status format. This field is loaded from the contents of the Receive Control register. Note that ERR, RNT, BRD, PRO, and AMC are configuration bits and are programmed during initialization. See Section 6.3.3 for the description of the Receive Control register.

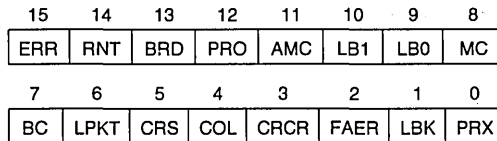


FIGURE 5-7. Receive Status Format

**byte count:** gives the length of the complete packet from the start of Destination Address to the end of Frame Check Sequence (FCS).

**packet pointer:** a 32-bit pointer that locates the packet in the RBA. The SONIC-T writes the contents of the CRBA0,1 registers into this field.

**sequence numbers:** this field displays the contents of two 8-bit counters (modulo 256) that sequence the RBAs used and the packets buffered. These counters assist the system in determining when an RBA has been completely processed. The sequence numbers allow the system to tally the packets that have been processed within a particular RBA. There are two sequence numbers that describe a packet: the RBA Sequence Number and the Packet Sequence Number. When a packet is buffered to memory, the SONIC-T maintains a single RBA Sequence Number for all packets in an RBA and sequences the Packet Number for succeeding packets in the RBA. When the SONIC-T uses the next RBA, it increments the RBA Sequence Number and clears the Packet Sequence Number. The RBA's sequence counter is not incremented when the Read RRA command is issued in the Command register. The format of the Receive Sequence Numbers is shown in Figure 5-8. These counters are reset during a SONIC-T hardware reset or by writing zero to them.

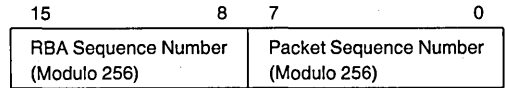


FIGURE 5-8. Receive Sequence Number Format

**receive link field:** a 15-bit pointer (A15-A1) that locates the next receive descriptor. The LSB of this field is the End Of List (EOL) bit and indicates the last descriptor in the list. (Initialized by the system.)

**In-use field:** this field provides a handshake between the system and the SONIC-T to indicate the ownership of the descriptor. When the system avails a descriptor to the SONIC-T, it writes a non-zero value into this field. The SONIC-T, in turn, sets this field to all "0's" when it has finished processing the descriptor. (That is, when the CRDA register has advanced to the next receive descriptor.) Generally, the SONIC-T releases control after writing the status and control information into the RDA. If, however, the SONIC-T has reached the last descriptor in the list, it maintains ownership of the descriptor until the system has appended additional descriptors to the list. The SONIC-T then relinquishes control after receiving the next packet. (See Section 5.4.6.1 for details on when the SONIC-T writes to this field.) The receive packet descriptor format is shown in Figure 5-6.

### 5.4.4 Receive Buffer Management Initialization

The Receive Resource, Descriptor, and Buffer areas (RRA, RDA, RBA) in memory and the appropriate SONIC-T registers must be properly initialized before the SONIC-T begins buffering packets. This section describes the initialization process.

#### 5.4.4.1 Initializing The Descriptor Page

All descriptor areas (RRA, RDA, and TDA) used by the SONIC-T reside within areas up to 32k (word) or 16k (long word) pages. This page may be placed anywhere within the 32-bit address range by loading the upper 16 address lines into the UTDA, URDA, and URRR registers.

#### 5.4.4.2 Initializing The RRA

The initialization of the RRA consists of loading the four SONIC-T RRA registers and writing the resource descriptor information to memory.

The RRA registers are loaded with the following values.

**Resource Start Area (RSA) register:** The RSA is loaded with the lower 16-bit address of the beginning of the RRA.

**Resource End Area (REA) register:** The REA is loaded with the lower 16-bit address of the end of the RRA. The end of the RRA is defined as the address of the last RXrsrc.ptr0 field in the RRA plus 4 words in 16-bit mode or 4 long words in 32-bit mode (Figure 5-4).

**Resource Read Pointer (RRP) register:** The RRP is loaded with the lower 16-bit address of the first resource descriptor the SONIC-T reads.

**Resource Write Pointer (RWP) register:** The RWP is loaded with the lower 16-bit address of the next vacant location where a resource descriptor will be placed by the system.

**Note:** The RWP register must only point to either (1) the RXrsrc.ptr0 field of one of the RRA Descriptors, (2) the memory address that the RSA points to (the start of the RRA), or (3) the memory address that the REA points to (the end of the RRA). When the RWP = RRP comparison is made, it is performed after the complete RRA descriptor has been read and not during the fetch. Failure to set the RWP to any of the above values prevents the RWP = RRP comparison from ever becoming true.



## 5.0 Buffer Management (Continued)

All RRA registers are concatenated with the URRRA register for generating the full 32-bit address.

The resource descriptors that the system writes to the RRA consists of four fields: (1) RXrsrc.buf\_ptr0, (2) RXrsrc.buf\_ptr1, (3) RXrsrc.buf\_wc0, and (4) RXrsrc.buf\_wc1. The fields must be contiguous (they cannot straddle the end points) and are written in the order shown in Figure 5-9. The "0" and "1" in the descriptors denote the least and most significant portions for the Buffer Pointer and Word Count. The first two fields supply the 32-bit starting location of the Receive Buffer Area (RBA), and the second two define the number of 16-bit words that the RBA occupies.

Note that two restrictions apply to the Buffer Pointer and Word Count. First, in 32-bit mode, since the SONIC-T always writes long words, an even count must be written to RXrsrc.buf\_wc0. Second, the Buffer Pointer must either be pointing to a word boundary in 16-bit mode (A0=0) or a long word boundary in 32-bit mode (A0,A1=0,0). Note also that the descriptors must be properly aligned in the RRA as discussed in Section 5.3.

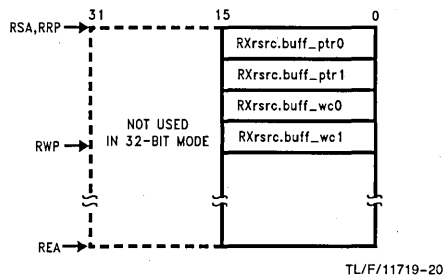


FIGURE 5-9. RRA Initialization

After configuring the RRA, the RRA Read command (setting RRRA bit in the Command register) may be given. This command causes the SONIC-T to read the RRA descriptor in a single block operation, and load the following registers (see Section 6.2 for register mnemonics):

- CRBA0 register ← RXrsrc.buf\_ptr0
- CRBA1 register ← RXrsrc.buf\_ptr1
- RBWC0 register ← RXrsrc.buf\_wc0
- RBWC1 register ← RXrsrc.buf\_wc1

When the command has completed, the RRRA bit in the Command register is reset to "0". Generally this command is only issued during initialization. At all other times, the RRA is automatically read as the SONIC-T finishes using an RBA.

### 5.4.4.3 Initializing The RDA

To accept multiple packets from the network, the receive packet descriptors must be linked together via the RXpkt.link fields. Each link field must be written with a 15-bit (A15-A1) pointer to locate the beginning of the next descriptor in the list. The LSB of the RXpkt.link field is the End Of List (EOL) bit and is used to indicate the end of the descriptor list. EOL = 1 for the last descriptor and EOL = 0 for the first or middle descriptors. The RXpkt.in\_use field indicates whether the descriptor is owned by the SONIC-T. The system writes a non-zero value to this field when the descriptor is available, and the SONIC-T writes all "0's" when it finishes using the descriptor. At startup, the Current Receive Descriptor Address (CRDA) register must be loaded with the address of the first RXpkt.status field in order for

the SONIC-T to begin receive processing at the first descriptor. An example of two descriptors linked together is shown in Figure 5-10. The fields initialized by the system are displayed in **bold type**. The other fields are written by the SONIC-T after a packet is accepted. The RXpkt.in\_use field is first written by the system, and then by the SONIC-T. Note that the descriptors must be aligned properly as discussed in Section 5.3. Also note that the URDA register is concatenated with the CRDA register to generate the full 32-bit address.

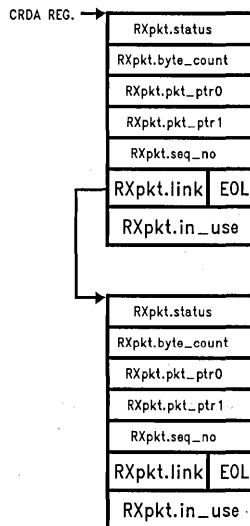


FIGURE 5-10. RDA Initialization Example

### 5.4.4.4 Initializing the Lower Boundary of the RBA

A "false bottom" is set in the RBA by loading the End Of Buffer Count (EOBC) register with a value equal to the maximum size packet in words (16 bits) that may be received. This creates a lower boundary in the RBA. Whenever the Remaining Buffer Word Count (RBWC0,1) registers decrement below the EOBC register, the SONIC-T buffers the next packet into another RBA. This also guarantees that a packet is always contiguously buffered into a single Receive Buffer Area (RBA). The SONIC-T does not buffer a packet into multiple RBAs. Note that in 32-bit mode, the SONIC-T holds the LSB always low so that it properly compares with the RBWC0,1 registers.

After a hardware reset, the EOBC register is automatically initialized to 2F8h (760 words or 1520 bytes). For 32-bit applications this is the suggested value for EOBC. EOBC defaults to 760 words (1520 bytes) instead of 759 words (1518 bytes) because 1518 is not a double word (32-bit) boundary (see Section 5.4.2.1). If the SONIC-T is used in 16-bit mode, then EOBC should be set to 759 words (1518 bytes) because 1518 is a word (16-bit) boundary.

Sometimes it may be desired to buffer a single packet per RBA. When doing this, it is important to set EOBC and the buffer size correctly. The suggested practice is to set EOBC to a value that is at least 4 bytes, in 32-bit mode, or 2 bytes, in 16-bit mode, less than the buffer size. An example of this for 32-bit mode is to set EOBC to 760 words (1520 bytes)



## 5.0 Buffer Management (Continued)

and the buffer size to 762 words (1524 bytes). A similar example for 16-bit mode would be EOBC = 759 words (1518 bytes) and the buffer size set to 760 words (1520 bytes). The buffer can be any size, but as long as the EOBC is 2 words, for 32-bit mode, or 1 word, for 16-bit mode, less than the buffer size, only one packet will be buffered in that RBA.

**Note 1:** It is possible to filter out most oversized packets by setting the buffer size to 760 words (1520 bytes) in 32-bit mode or 759 words (1518 bytes) in 16-bit mode. EOBC would be set to 758 words (1516 bytes) for both cases. With this configuration, any packet over 1520 bytes, in 32-bit mode, or 1518 bytes, in 16-bit mode, will not be completely buffered because the packet will overflow the buffer. When a packet overflow occurs, a Receive Buffer Area Exceeded interrupt (RBAE in the Interrupt Status Register, Section 6.3.6) will occur.

**Note 2:** When buffering one packet per buffer, it is suggested that the values in Note 1 above be used. Since the minimum legal sized Ethernet packet is 64 bytes, however, it is possible to set EOBC as much as 64 bytes less than the buffer size and still end up with one packet per buffer. Figure 5-11 shows this "range."

### 5.4.5 Beginning Of Reception

At the beginning of reception, the SONIC-T checks its internally stored EOL bit from the previous RXpkt.link field for a "1". If the SONIC-T finds EOL = 1, it recognizes that after the previous reception, there were no more remaining receive packet descriptors. It re-reads the same RXpkt.link field to check if the system has updated this field since the last reception. If the SONIC-T still finds EOL = 1, reception ceases. (See Section 5.5 for adding descriptors to the list.) Otherwise, the SONIC-T begins storing the packet in the RBA starting at the Current Receive Buffer Address (CRBA0,1) registers and continues until the packet has completed. Concurrent with the packet reception, the Remaining Buffer Word Count (RBWC0,1) registers are decremented after each word is written to memory. This register determines the remaining words in the RBA at the end of reception.

### 5.4.6 End Of Packet Processing

At the end of a reception, the SONIC-T enters its end of packet processing sequence to determine whether to accept or reject the packet based on receive errors and packet size. At the end of reception the SONIC-T enters one of the following two sequences:

- Successful reception sequence
- Buffer recovery for runt packets or packets with errors

### 5.4.6.1 Successful Reception

If the SONIC-T accepts the packet, it first writes 5 words of descriptor information in the RDA beginning at the address pointed to by the Current Receive Descriptor Address (CRDA) register. It then reads the RXpkt.link field to advance the CRDA register to the next receive descriptor. The SONIC-T also checks the EOL bit for a "1" in this field. If EOL = 1, no more descriptors are available for the SONIC-T. The SONIC-T recovers the address of the current RXpkt.link field (from a temporary register) and generates a "Receive Descriptors Exhausted" indication in the Interrupt Status register. (See Section 5.4.7 on how to add descriptors.) The SONIC-T maintains ownership of the descriptor by *not* writing to the RXpkt.in\_use field. Otherwise, if EOL = 0, the SONIC-T advances the CRDA register to the next descriptor and resets the RXpkt.in\_use field to all "0's".

The SONIC-T accesses the complete 7 word RDA descriptor in a single block operation.

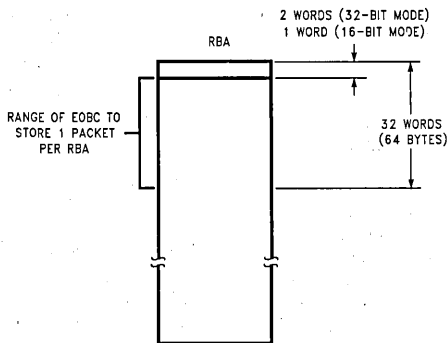
The SONIC-T also checks if there is remaining space in the RBA. The SONIC-T compares the Remaining Buffer Word Count (RBWC0,1) registers with the static End Of Buffer Count (EOBC). If the RBWC is less than the EOBC, a maximum sized packet will no longer fit in the remaining space in the RBA; hence, the SONIC-T fetches a resource descriptor from the RRA and loads its registers with the pointer and word count of the next available RBA.

### 5.4.6.2 Buffer Recovery For Runt Packets Or Packets With Errors

If a runt packet (less than 64 bytes) or packet with errors arrives and the Receive Control register has been configured to not accept these packets, the SONIC-T recovers its pointers back to the original positions. The CRBA0,1 registers are not advanced and the RBWC0,1 registers are not decremented. The SONIC-T recovers its pointers by maintaining a copy of the buffer address in the Temporary Receive Buffer Address registers (TRBA0,1). The SONIC-T recovers the value in the RBWC0,1 registers from the Temporary Buffer Word Count registers (TBWC0,1).

### 5.4.7 Overflow Conditions

When an overflow condition occurs, the SONIC-T halts its DMA operations to prevent writing into unauthorized memory. The SONIC-T uses the Interrupt Status register (ISR) to indicate three possible overflow conditions that can occur



$$\text{Range of EOBC} = (\text{RXsrc.wc0,1} - 2 \text{ to } \text{RXsrc.wc0,1} - 32)$$

**FIGURE 5-11. Setting EOBC for Single Packet RBA**

TL/F/11719-22

## 5.0 Buffer Management (Continued)

when its receive resources have been exhausted. The system should respond by replenishing the resources that have been exhausted. These overflow conditions (Descriptor Resources Exhausted, Buffer Resources Exhausted, and RBA Limit Exceeded) are indicated in the Interrupt Status register and are detailed as follows:

**Descriptor Resources Exhausted:** This occurs when the SONIC-T has reached the last receive descriptor in the list, meaning that the SONIC-T has detected EOL = 1. The system must supply additional descriptors for continued reception. The system can do this in one of two ways: 1) appending descriptors to the existing list, or 2) creating a separate list.

1. Appending descriptors to the existing list. This is the easiest and preferred way. To do this, the system, after creating the new list, joins the new list to the existing list by simply writing the beginning address of the new list into the RXpkt.link field and setting EOL = 0. At the next reception, the SONIC-T re-reads the last RXpkt.link field, and updates its CRDA register to point to the next descriptor.

2. Creating a separate list. This requires an additional step because the lists are not joined together and requires that the CRDA register be loaded with the address of the RXpkt.link field in the new list.

During this overflow condition, the SONIC-T maintains ownership of the descriptor (RXpkt.in\_use  $\neq$  00h) and waits for the system to add additional descriptors to the list. When the system appends more descriptors, the SONIC-T releases ownership of the descriptor after writing 0000h to the RXpkt.in\_use field.

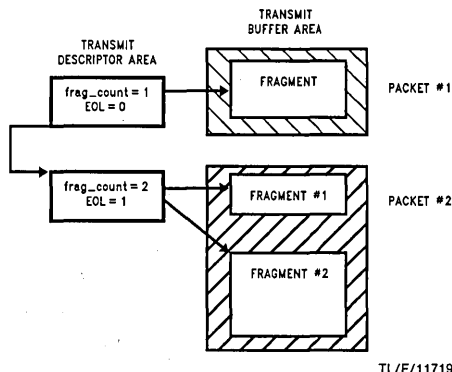
**Buffer Resources Exhausted:** This occurs when the SONIC-T has detected that the Resource Read Pointer (RRP) and Resource Write Pointer (RWP) registers are equal (i.e., all RRA descriptors have been exhausted). The RBE bit in the Interrupt Status register is set when the SONIC-T finishes using the second to last receive buffer and reads the last RRA descriptor. Actually, the SONIC-T is not truly out of resources, but gives the system an early warning of an impending out of resources condition. To continue reception after the last RBA is used, the system must supply additional RRA descriptor(s), update the RWP register, and clear the RBE bit in the ISR. The SONIC-T rereads the RRA after this bit is cleared.

**RBA Limit Exceeded:** This occurs when a packet does not completely fit within the remaining space of the RBA. This can occur if the EOBC register is not programmed to a value greater than the largest packet that can be received. When this situation occurs, the packet is truncated and the SONIC-T reads the RRA to obtain another RBA. Indication of an RBA limit being exceeded is signified by the Receive Buffer Area Exceeded (RBAE) interrupt being set (see Section 6.3.6). An RDA will not be set up for the truncated packet and the buffer space will not be re-used. To rectify this potential overflow condition, the EOBC register must be loaded with a value equal to or greater than the largest packet that can be accepted. (See Section 5.4.2.)

### 5.5 TRANSMIT BUFFER MANAGEMENT

To begin transmission, the system software issues the Transmit command (TXP = 1 in the CR). The Transmit Buffer Management uses two areas in memory for transmitting packets (Figure 5-12), the Transmit Descriptor Area (TDA)

and the Transmit Buffer Area (TBA). During transmission, the SONIC-T fetches control information from the TDA, loads its appropriate registers, and then transmits the data from the TBA. When the transmission is complete, the SONIC-T writes the status information in the TDA. From a single transmit command, packets can either be transmitted singly or in groups if several descriptors have been linked together.



TL/F/11719-23

FIGURE 5-12. Overview of Transmit Buffer Management

#### 5.5.1 Transmit Descriptor Area (TDA)

The TDA contains descriptors that the system has generated to exchange status and control information. Each descriptor corresponds to a single packet and consists of the following 16-bit fields.

**TXpkt.status:** This field is written by the SONIC-T and provides status of the transmitted packet. (See Section 5.5.1.2 for more details.)

**TXpkt.config:** This field allows programming the SONIC-T to one of the various transmit modes. The SONIC-T reads this field and loads the corresponding configuration bits (PINTR, POWC, CRCI, and EXDIS) into the Transmit Control register. (See Section 5.5.1.1 for more details.)

**TXpkt.pkt\_size:** This field contains the byte count of the entire packet.

**TXpkt.frag\_count:** This field contains the number of fragments the packet is segmented into.

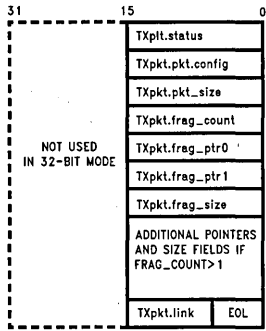
**TXpkt.frag\_ptr0,1:** This field contains a 32-bit pointer which locates the packet fragment to be transmitted in the Transmit Buffer Area (TBA). This pointer is not restricted to any byte alignment.

**TXpkt.frag\_size:** This field contains the byte count of the packet fragment. The minimum fragment size is 1 byte.

**TXpkt.link:** This field contains a 15-bit pointer (A15-A1) to the next TDA descriptor. The LSB, the End Of List (EOL) bit, indicates the last descriptor in the list when set to a "1". When descriptors have been linked together, the SONIC-T transmits back-to-back packets from a single transmit command.

The data of the packet does not need to be contiguous, but can exist in several locations (fragments) in memory. In this case, the TXpkt.frag\_count field is greater than one, and additional TXpkt.frag\_ptr0,1 and TXpkt.frag\_size fields corresponding to each fragment are used. The descriptor format is shown in Figure 5-13. Note that in 32-bit mode the upper word, D<31:16>, is not used.

## 5.0 Buffer Management (Continued)



TL/F/11719-24

FIGURE 5-13. Transmit Descriptor Area

### 5.5.1.1 Transmit Configuration

The TXpkt.config field allows the SONIC-T to be programmed into one of the transmit modes before each transmission. At the beginning of each transmission, the SONIC-T reads this field and loads the PINTR, POWC, CRCI, and EXDIS bits into the Transmit Control Register (TCR). The configuration bits in the TCR correspond directly with the bits in the TXpkt.config field as shown in Figure 5-14. See Section 6.3.4 for the description on the TCR.

15	14	13	12	11	10	9	8
PINTR	POWC	CRCI	EXDIS	X	X	X	X
7	6	5	4	3	2	1	0
X	X	X	X	X	X	X	X

Note: x = don't care

FIGURE 5-14. TXpkt.config Field

### 5.5.1.2 Transmit Status

At the end of each transmission the SONIC-T writes the status bits (<10:0>) of the Transmit Control Register (TCR) and the number of collisions experienced during the transmission into the TXpkt.status field (Figure 5-15, res = reserved). Bits NC4-NC0 indicate the number of collisions where NC4 is the MSB. See Section 6.3.4 for the description of the TCR.

15	14	13	12	11	10	9	8
NC4	NC3	NC2	NC1	NC0	EXD	DEF	NCRS
7	6	5	4	3	2	1	0
CRSL	EXC	OWC	res	PMB	FU	BCM	PTX

FIGURE 5-15. TXpkt.status Field

### 5.5.2 Transmit Buffer Area (TBA)

The TBA contains the fragments of packets that are defined by the descriptors in the TDA. A packet can consist of a single fragment or several fragments, depending upon the fragment count in the TDA descriptor. The fragments also can reside anywhere within the full 32-bit address range, and be aligned to any byte boundary. When an odd byte boundary is given, the SONIC-T automatically begins reading data at the corresponding word boundary in 16-bit mode or a long word boundary in 32-bit mode. The SONIC-T ig-

nores the extraneous bytes which are written into the FIFO during odd byte alignment fragments. The minimum allowed fragment size is 1 byte. Figure 5-12 shows the relationship between the TDA and the TBA for single and multi-fragmented packets.

### 5.5.3 Preparing To Transmit

All fields in the TDA descriptor and the Current Transmit Descriptor Address (CTDA) register of the SONIC-T must be initialized before the Transmit Command (setting the TXP bit in the Command register) can be issued. If more than one packet is queued, the descriptors must be linked together with the TXpkt.link field. The last descriptor must have EOL = 1 and all other descriptors must have EOL = 0. To begin transmission, the system loads the address of the first TXpkt.status field into the CTDA register. Note that the upper 16-bits of address are loaded in the Upper Transmit Descriptor (UTDA) register. The user performs the following transmit initialization.

1. Initialize the TDA
2. Load the CTDA register with the address of the first transmit descriptor
3. Issue the transmit command

Note that if the Source Address of the packet being transmitted is not in the CAM, the Packet Monitored Bad (PMB) bit in the TXpkt.status field will be set (see Section 6.3.4).

#### 5.5.3.1 Transmit Process

When the Transmit Command (TXP = 1 in the Command register) is issued, the SONIC-T fetches the control information in the TDA descriptor, loads its appropriate registers (shown below) and begins transmission. (See Section 6.2 for register mnemonics.)

- TCR ← TXpkt.config
- TPS ← TXpkt.pkt\_size
- TFC ← TXpkt.frag\_count
- TSA0 ← TXpkt.frag\_ptr0
- TSA1 ← TXpkt.frag\_ptr1
- TFS ← TXpkt.frag\_size
- CTDA ← TXpkt.link

(CTDA is loaded after all fragments have been read and successfully transmitted. If the halt transmit command is issued (HTX bit in the Command register is set) the CTDA register is not loaded.)

During transmission, the SONIC-T reads the packet descriptor in the TDA and transmits the data from the TBA. If TXpkt.frag\_count is greater than one, the SONIC-T, after finishing transmission of the fragment, fetches the next TXpkt.frag\_ptr0,1 and TXpkt.frag\_size fields and transmits the next fragment. This process continues until all fragments of a packet are transmitted. At the end of packet transmission, status is written in to the TXpkt.status field. The SONIC-T then reads the TXpkt.link field and checks if EOL = 0. If it is "0", the SONIC-T fetches the next descriptor and transmits the next packet. If EOL = 1 the SONIC-T generates a "Transmission Done" indication in the Interrupt Status register and resets the TXP bit in the Command register.

In the event of a collision, the SONIC-T recovers its pointer in the TDA and retransmits the packet up to 15 times. The SONIC-T maintains a copy of the CTDA register in the Temporary Transmit Descriptor Address (TTDA) register.

## 5.0 Buffer Management (Continued)

The SONIC-T performs a block operation of 6, 3, or 2 accesses in the TDA, depending on where the SONIC-T is in the transmit process. For the first fragment, it reads the TXpkt.config to TXpkt.frag\_size (6 accesses). For the next fragment, if any, it reads the next 3 fields from TXpkt.frag\_ptr0 to TXpkt.frag\_size (3 accesses). At the end of transmission it writes the status information to TXpkt.status and reads the TXpkt.link field (2 accesses).

### 5.5.3.2 Transmit Completion

The SONIC-T stops transmitting under two conditions. In the normal case, the SONIC-T transmits the complete list of descriptors in the TDA and stops after it detects EOL = 1. In the second case, certain transmit errors cause the SONIC-T to abort transmission. If *FIFO Underrun*, *Byte Count Mismatch*, *Excessive Collision*, or *Excessive Deferral* (if enabled) errors occur, transmission ceases. The CTDA register points to the last packet transmitted. The system can also halt transmission under software control by setting the HTX bit in the Command register. Transmission halts after the SONIC-T writes to the TXpkt.status field.

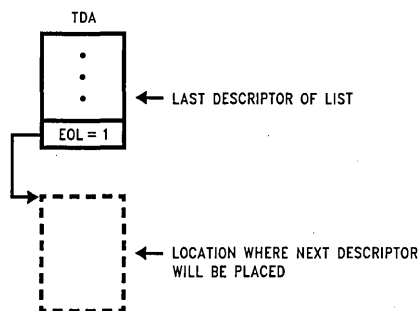
### 5.5.4 Dynamically Adding TDA Descriptors

Descriptors can be dynamically added during transmission without halting the SONIC-T. The SONIC-T can also be guaranteed to transmit the complete list including newly appended descriptors (barring any transmit abort conditions) by observing the following rule: The last TXpkt.link field must point to the next location where a descriptor will be added (see step 3 below and *Figure 5-16*).

The procedure for appending descriptors consists of:

1. Creating a new descriptor with its TXpkt.link pointing to the next vacant descriptor location and its EOL bit set to a "1".
2. Resetting the EOL bit to a "0" of the previously last descriptor.
3. Re-issuing the Transmit command (setting the TXP bit in the Command register).

Step 3 assures that the SONIC-T will transmit all the packets in the list. If the SONIC-T is currently transmitting, the Transmit command has no effect and continues transmitting until it detects EOL = 1. If the SONIC-T had just finished transmitting, it continues transmitting from where it had previously stopped.



TL/F/11719-25

FIGURE 5-16. Initializing Last Link Field

## 6.0 SONIC-T Registers

The SONIC-T contains two sets of registers: The status/control registers and the CAM memory cells. The status/control registers are used to configure, control, and monitor SONIC-T operation. They are directly addressable registers and occupy 64 consecutive address locations in the system memory space (selected by the RA5-RA0 address pins). There are a total of 64 status/control registers divided into the following categories:

**User Registers:** These registers are accessed by the user to configure, control, and monitor SONIC-T operation. These are the only SONIC-T registers the user needs to access. *Figure 6-3* shows the programmer's model and *Table 6-1* lists the attributes of each register.

**Internal Use Registers:** These registers (*Table 6-2*) are used by the SONIC-T during normal operation and are not intended to be accessed by the user.

**National Factory Test Registers:** These registers (*Table 6-3*) are for National factory use only and should never be accessed by the user. Accessing these registers during normal operation can cause improper functioning of the SONIC-T.

### 6.1 THE CAM UNIT

The CAM unit memory cells are indirectly accessed by programming the CAM descriptor area in system memory and issuing the LCAM command (setting the LCAM bit in the Control register). The CAM cells do not occupy address locations in register space and, thus, are not accessible through the RA5-RA0 address pins. The CAM control registers, however, are part of the user register set and must be initialized before issuing the LCAM command (see *Section 6.3.10*).

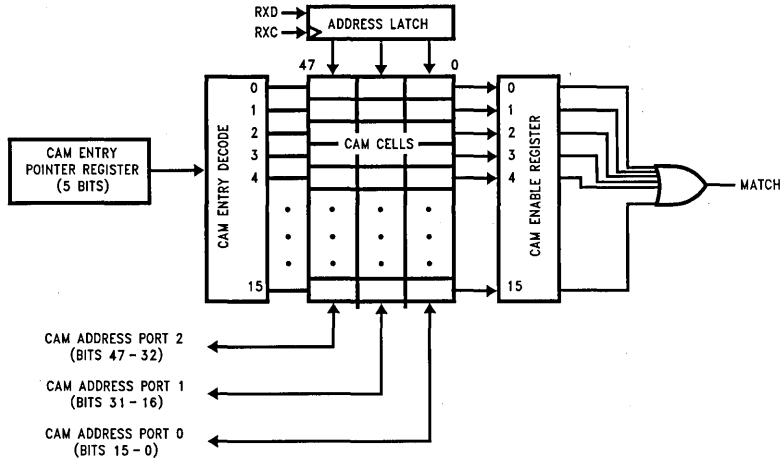
The Content Addressable Memory (CAM) consists of sixteen 48-bit entries for complete address filtering (*Figure 6-1*) of network packets. Each entry corresponds to a 48-bit destination address that is user programmable and can contain any combination of Multicast or Physical addresses. Each entry is partitioned into three 16-bit CAM cells accessible through CAM Address Ports (CAP2, CAP1 and CAP0) with CAP0 corresponding to the least significant 16 bits of the Destination Address and CAP2 corresponding to the most significant bits. The CAM is accessed in a two step process. First, the CAM Entry Pointer is loaded to point to one of the 16 entries. Then, each of the CAM Address Ports is accessed to select the CAM cell. The 16 user programmable CAM entries can be masked out with the CAM Enable register (see *Section 6.3.10*).

**Note:** It is not necessary to program a broadcast address into the CAM when it is desired to accept broadcast packets. Instead, to accept broadcast packets, set the BRD bit in the Receive Control register. If the BRD bit has been set, the CAM is still active. This means that it is possible to accept broadcast packets at the same time as accepting packets that match physical addresses in the CAM.

#### 6.1.1 The Load CAM Command

Because the SONIC-T uses the CAM for a relatively long period of time during reception, it can only be written to via the CAM Descriptor Area (CDA) and is only readable when

## 6.0 SONIC-T Registers (Continued)



TL/F/11719-26

FIGURE 6-1. CAM Organization

the SONIC-T is in software reset. The CDA resides in the same 64k byte block of memory as the Receive Resource Area (RRA) and contains descriptors for loading the CAM registers. These descriptors are contiguous and each descriptor consists of four 16-bit fields (Figure 6-2). In 32-bit mode the upper word,  $D < 31:16 >$ , is not used. The first field contains the value to be loaded into the CAM Entry Pointer and the remaining fields are for the three CAM Address Ports (see Section 6.3.10). In addition, there is one more field after the last descriptor containing the mask for the CAM Enable register. Each of the CAM descriptors are addressed by the CAM Descriptor Pointer (CDP) register.

After the system has initialized the CDA, it can issue the Load CAM command to program the SONIC-T to read the CDA and load the CAM. The procedure for issuing the Load CAM command is as follows.

1. Initialize the Upper Receive Resource Address (URRA) register. Note that the CAM Descriptor Area must reside within the same 64k page as the Receive Resource Area. (See Section 6.3.9.)

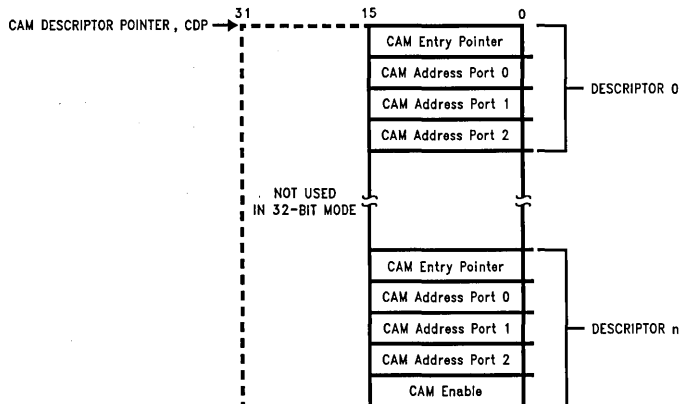
2. Initialize the CDA as described above.

3. Initialize the CAM Descriptor Count with the number of CAM descriptors. Note, only the lower 5 bits are used in this register. The other bits are don't cares. (See Section 6.3.10.)

4. Initialize the CAM Descriptor Pointer to locate the first descriptor in the CDA. This register must be reloaded each time a new Load CAM command is issued.

5. Issue the Load CAM command (LCAM) in the Command register. (See Section 6.3.1.)

If a transmission or reception is in progress, the CAM DMA function will not occur until these operations are complete. When the SONIC-T completes the Load CAM command, the CDP register points to the next location after the CAM enable field and the CDC equals zero. The SONIC-T resets the LCAM bit in the Command register and sets the Load CAM Done (LCD) bit in the ISR.



TL/F/11719-27

FIGURE 6-2. CAM Descriptor Area Format

## 6.0 SONIC-T Registers (Continued)

	RA <5:0>	15	0
Status and Control Registers	0h Command Register	Status and Control Fields	
	1 Data Configuration Register	Control Fields	
	2 Receive Control Register	Status and Control Fields	
	3 Transmit Control Register	Status and Control Fields	
	4 Interrupt Mask Register	Mask Fields	
Transmit Registers	5 Interrupt Status Register	Status Fields	
	3F Data Configuration Register 2	Control Fields	
	6 Upper Transmit Descriptor Address Register	Upper 16-bit Address Base	
Receive Registers	7 Current Transmit Descriptor Address Register	Lower 16-bit Address Offset	
	0D Upper Receive Descriptor Address Register	Upper 16-bit Address Base	
	0E Current Receive Descriptor Address Register	Lower 16-bit Address Offset	
	14 Upper Receive Resource Address Register	Upper 16-bit Address Base	
	15 Resource Start Address Register	Lower 16-bit Address Offset	
	16 Resource End Address Register	Lower 16-bit Address Offset	
	17 Resource Read Register	Lower 16-bit Address Offset	
	18 Resource Write Register	Lower 16-bit Address Offset	
	2B Receive Sequence Counter	Count Value	8   7   Count Value
CAM Registers	21 CAM Entry Pointer	4	
	22 CAM Address Port 2	Most Significant 16 bits of CAM Entry	
	23 CAM Address Port 1	Middle 16 bits of CAM Entry	
	24 CAM Address Port 0	Least Significant 16 bits of CAM Entry	
	25 CAM Enable Register	Mask Fields	
	26 CAM Descriptor Pointer	Lower 16-bit Address Offset	
Tally Counters	27 CAM Descriptor Count	5   Count Value	
	2C DRC Error Tally Counter	Count Value	
	2D Frame Alignment Error Tally	Count Value	
Watchdog Timer	2E Missed Packet Tally	Count Value	
	29 Watchdog Timer 0	Lower 16-bit Count Value	
	2A Watchdog Timer 1	Upper 16-bit Count Value	
	28 Silicon Revision Register	Chip Revision Number	

FIGURE 6-3. SONIC-T Register Programming Model

## 6.0 SONIC-T Registers (Continued)

### 6.2 STATUS/CONTROL REGISTERS

This set of registers is used to convey status/control information to/from the host system and to control the operation of the SONIC-T. These registers are used for loading commands generated from the system, indicating transmit and receive status, buffering data to/from memory, and provid-

ing interrupt control. The registers are selected by asserting chip select to the SONIC-T and providing the necessary address on register address pins RA5-RA0. Tables 6-1, 6-2, and 6-3 show the locations of all SONIC-T registers and where information on the registers can be found in the data sheet.

**TABLE 6-1. User Registers**

RA5-RA0	Access	Register	Symbol	Description (section)
<b>COMMAND AND STATUS REGISTERS</b>				
00h	R/W	Command	CR	6.3.1
01 (Note 3)	R/W	Data Configuration	DCR	6.3.2
02	R/W	Receive Control	RCR	6.3.3
03	R/W	Transmit Control	TCR	6.3.4
04	R/W	Interrupt Mask	IMR	6.3.5
05	R/W	Interrupt Status	ISR	6.3.6
3F (Note 3)	R/W	Data Configuration 2	DCR2	6.3.7
<b>TRANSMIT REGISTERS</b>				
06	R/W	Upper Transmit Descriptor Address	UTDA	6.3.8, 5.4.4.1
07	R/W	Current Transmit Descriptor Address	CTDA	6.3.8, 5.5.3
<b>RECEIVE REGISTERS</b>				
0D	R/W	Upper Receive Descriptor Address	URDA	6.3.9, 5.4.4.1
0E	R/W	Current Receive Descriptor Address	CRDA	6.3.9, 5.4.4.3
13	R/W	End of Buffer Word Count	EOBC	6.3.9, 5.4.2
14	R/W	Upper Receive Resource Address	URRA	6.3.9, 5.4.4.1
15	R/W	Resource Start Address	RSA	6.3.9, 5.4.1
16	R/W	Resource End Address	REA	6.3.9, 5.4.1
17	R/W	Resource Read Pointer	RRP	6.3.9, 5.4.1
18	R/W	Resource Write Pointer	RWP	6.3.9, 5.4.1
2B	R/W	Receive Sequence Counter	RSC	6.3.9, 5.4.3.2
<b>CAM REGISTERS</b>				
21	R/W	CAM Entry Pointer	CEP	6.1, 6.3.10
22 (Note 1)	R	CAM Address Port 2	CAP2	6.1, 6.3.10
23 (Note 1)	R	CAM Address Port 1	CAP1	6.1, 6.3.10
24 (Note 1)	R	CAM Address Port 0	CAP0	6.1, 6.3.10
25 (Note 2)	R/W	CAM Enable	CE	6.1, 6.3.10
26	R/W	CAM Descriptor Pointer	CDP	6.1, 6.3.10
27	R/W	CAM Descriptor Count	CDC	6.1, 6.3.10
<b>TALLY COUNTERS</b>				
2C (Note 4)	R/W	CRC Error Tally	CRCT	6.3.11
2D (Note 4)	R/W	FAE Tally	FAET	6.3.11
2E (Note 4)	R/W	Missed Packet Tally	MPT	6.3.11



## 6.0 SONIC-T Registers (Continued)

**TABLE 6-1. User Registers (Continued)**

RA5-RA0	Access	Register	Symbol	Description (section)
<b>WATCHDOG COUNTERS</b>				
29	R/W	Watchdog Timer	WT0	6.3.12
2A	R/W	Watchdog Timer 1	WT1	6.3.12
<b>SILICON REVISION</b>				
28	R	Silicon Revision	SR	6.3.13

**Note 1:** These registers can only be read when the SONIC-T is in reset mode (RST bit in the CR is set). The SONIC-T gives invalid data when these registers are read in non-reset mode.

**Note 2:** This register can only be written to when the SONIC-T is in reset mode. This register is normally only loaded by the Load CAM command.

**Note 3:** The Data Configuration registers, DCR and DCR2, can only be written to when the SONIC-T is in reset mode (RST bit in CR is set). Writing to these registers while not in reset mode does not alter the registers.

**Note 4:** The data written to these registers is inverted before being latched. That is, if a value of FFFFh is written, these registers will contain and read back the value of 0000h. Data is not inverted during a read operation.

**TABLE 6-2. Internal Use Registers (Users should not write to these registers)**

(RA5-RA0)	Access	Register	Symbol	Description (section)
<b>TRANSMIT REGISTERS</b>				
08 (Note 1)	R/W	Transmit Packet Size	TPS	5.5
09	R/W	Transmit Fragment Count	TFC	5.5
0A	R/W	Transmit Start Address 0	TSA0	5.5
0B	R/W	Transmit Start Address 1	TSA1	5.5
0C (Note 2)	R/W	Transmit Fragment Size	TFS	5.5
20	R/W	Temporary Transmit Descriptor Address	TTDA	5.5.4
2F	R	Maximum Deferral Timer	MDT	6.3.4
<b>RECEIVE REGISTERS</b>				
0F	R/W	Current Receive Buffer Address 0	CRBA0	5.4.2, 5.4.4.2
10	R/W	Current Receive Buffer Address 1	CRBA1	5.4.2, 5.4.4.2
11	R/W	Remaining Buffer Word Count 0	RBWC0	5.4.2, 5.4.4.2
12	R/W	Remaining Buffer Word Count 1	RBWC1	5.4.2, 5.4.4.2
19	R/W	Temporary Receive Buffer Address 0	TRBA0	5.4.6.2
1A	R/W	Temporary Receive Buffer Address 1	TRBA1	5.4.6.2
1B	R/W	Temporary Buffer Word Count 0	TBWC0	5.4.6.2
1C	R/W	Temporary Buffer Word Count 1	TBWC1	5.4.6.2
1F	R/W	Last Link Field Address	LLFA	none
<b>ADDRESS GENERATORS</b>				
1D	R/W	Address Generator 0	ADDR0	none
1E	R/W	Address Generator 1	ADDR1	none

**Note 1:** The data that is read from these registers is the inversion of what has been written to them.

**Note 2:** The value that is written to this register is shifted once in 16-bit mode and shifted twice in 32-bit mode.

**TABLE 6-3. National Factory Test Registers (Users should not access these registers)**

(RA5-RA0)	Access	Register	Symbol	Description (section)
30 • 3E	R/W	These registers are for factory use only. Users must not address these registers or improper SONIC-T operation can occur.	none	none

## 6.0 SONIC-T Registers (Continued)

### 6.3 REGISTER DESCRIPTION

#### 6.3.1 Command Register

(RA <5:0> = 0h)

This register (Figure 6-4) is used for issuing commands to the SONIC-T. These commands are issued by setting the corresponding bits for the function. For all bits, except for the RST bit, the SONIC-T resets the bit after the command is completed. With the exception of RST, writing a "0" to any bit has no effect. Before any commands can be issued, the RST bit must first be reset to "0". This means that, if the RST bit is set, two writes to the Command Register are required to issue a command to the SONIC-T; one to clear the RST bit, and one to issue the command.

This register also controls the general purpose 32-bit Watchdog Timer. After the Watchdog Timer register has been loaded, it begins to decrement once the ST bit has been set to "1". An interrupt is issued when the count reaches zero if the Timer Complete interrupt is enabled in the IMR.

During hardware reset, bits 7, 4, and 2 are set to a "1"; all others are cleared. During software reset bits 9, 8, 1, and 0 are cleared and bits 7 and 2 are set to a "1"; all others are unaffected.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	LCAM	RRRA	RST	0	ST	STP	RXEN	RXDIS	TXP	HTX
						r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

r = read only, r/w = read/write

**FIGURE 6-4. Command Register**

Field	Meaning
LCAM	LOAD CAM
RRRA	READ RRA
RST	SOFTWARE RESET
ST	START TIMER
STP	STOP TIMER
RXEN	RECEIVER ENABLE
RXDIS	RECEIVER DISABLE
TXP	TRANSMIT PACKET(S)
HTX	HALT TRANSMISSION

Bit	Description
15-10	<b>Must be 0</b>
9	<p><b>LCAM: LOAD CAM</b> Setting this bit causes the SONIC-T to load the CAM with the descriptor that is pointed to by the CAM Descriptor Pointer register.</p> <p><i>Note:</i> This bit must not be set during transmission (TXP is set). The SONIC-T will lock up if both bits are set simultaneously.</p>
8	<p><b>RRRA: READ RRA</b> Setting this bit causes the SONIC-T to read the next RRA descriptor pointed to by the Resource Read Pointer (RRP) register. Generally this bit is only set during initialization. Setting this bit during normal operation can cause improper receive operation.</p>
7	<p><b>RST: SOFTWARE RESET</b> Setting this bit resets all internal state machines. The CRC generator is disabled and the Tally counters are halted, but not cleared. The SONIC-T becomes operational when this bit is reset to "0". A hardware reset sets this bit to a "1". It must be reset to "0" before the SONIC-T becomes operational.</p>
6	<b>Must be 0</b>
5	<p><b>ST: START TIMER</b> Setting this bit enables the general-purpose watchdog timer to begin counting or to resume counting after it has been halted. This bit is reset when the timer is halted (i.e., STP is set). Setting this bit resets STP.</p>
4	<p><b>STP: STOP TIMER</b> Setting this bit halts the general-purpose watchdog timer and resets the ST bit. The timer resumes when the ST bit is set. This bit powers up as a "1". <i>Note:</i> Simultaneously setting bits ST and STP stops the timer.</p>

## 6.0 SONIC-T Registers (Continued)

### 6.3.1 Command Register (Continued)

Bit	Description
3	<p><b>RXEN: RECEIVER ENABLE</b></p> <p>Setting this bit enables the receive buffer management engine to begin buffering data to memory. Setting this bit resets the RXDIS bit. Note: If this bit is set while the MAC unit is currently receiving a packet, both RXEN and RXDIS are set until the network goes inactive (i.e., the SONIC-T will not start buffering in the middle of a packet being received). When both RXEN and RXDIS are set, RXEN could be cleared by writing zero to it.</p>
2	<p><b>RXDIS: RECEIVER DISABLE</b></p> <p>Setting this bit disables the receiver from buffering data to memory or the Receive FIFO. If this bit is set during the reception of a packet, the receiver is disabled only after the packet is processed. The RXEN bit is reset when the receiver is disabled. Tally counters remain active regardless of the state of this bit.</p> <p><b>Note:</b> If this bit is set while the SONIC-T is currently receiving a packet, both RXEN and RXDIS are set until the packet is fully received. When both RXEN and RXDIS are set, RXDIS could be cleared by writing zero to it.</p>
1	<p><b>TXP: TRANSMIT PACKET(S)</b></p> <p>Setting this bit causes the SONIC-T to transmit packets which have been set up in the Transmit Descriptor Area (TDA). The SONIC-T loads its appropriate registers from the TDA, then begins transmission. The SONIC-T clears this bit after any of the following conditions have occurred: (1) transmission had completed (i.e., after the SONIC-T has detected EOL = 1), (2) the Halt Transmission command (HTX) has taken effect, or (3) a transmit abort condition has occurred. This condition occurs when any of the following bits in the TCR have been set: EXC, EXD, FU, or BCM. This bit must not be set if a Load CAM operation is in progress (LCAM is set). The SONIC-T will lock up if both bits are set simultaneously.</p>
0	<p><b>HTX: HALT TRANSMISSION</b></p> <p>Setting this bit halts the transmit command after the current transmission has completed. TXP is reset after transmission has halted. The Current Transmit Descriptor Address (CTDA) register points to the last descriptor transmitted. The SONIC-T samples this bit after writing to the Txpkt.status field.</p>

## 6.0 SONIC-T Registers (Continued)

### 6.3.2 Data Configuration Register

(RA<5:0> = 1h)

This register (*Figure 6-5*) establishes the bus cycle options for reading/writing data to/from 16- or 32-bit memory systems. During a hardware reset, bits 15 and 13 are cleared; all other bits are unaffected. (Because of this, the first thing the driver software does to the SONIC-T should be to set up this register.) All bits are unaffected by a software reset. This register must only be accessed when the SONIC-T is in reset mode (i.e., the RST bit is set in the Command register).

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	EXBUS	0	LBR	PO1	PO0	SBUS	USR1	USR0	WC1	WC0	DW	BMS	RFT1	RFT0	TFT1	TFT0
	r/w		r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

r = read only, r/w = read/write

**FIGURE 6-5. Data Configuration Register**

Field	Meaning
EXBUS	EXTENDED BUS MODE
LBR	LATCHED BUS RETRY
PO0,PO1	PROGRAMMABLE OUTPUTS
SBUS	SYNCHRONOUS BUS MODE
USR0, USR1	USER DEFINABLE PINS
WC0, WC1	WAIT STATE CONTROL
DW	DATA WIDTH SELECT
BMS	BLOCK MODE SELECT FOR DMA
RFT0, RFT1	RECEIVE FIFO THRESHOLD
TFT0, TFT1	TRANSMIT FIFO THRESHOLD

Bit	Description
15	<p><b>EXBUS: EXTENDED BUS MODE</b></p> <p>Setting this bit enables the Extended Bus mode which enables the following:</p> <ol style="list-style-type: none"> <li>1. Extended Programmable Outputs, EXUSR &lt;3:0&gt;: This changes the TXD, LBK, RXC and RXD pins from the external ENDEC interface into four programmable user outputs, EXUSR &lt;3:0&gt; respectively, which are similar to USR &lt;1:0&gt;. These outputs are programmed with bits 15-12 in the DCR2 (see Section 6.3.7). On hardware reset, these four pins will be TRI-STATE and will remain that way until the DCR is changed. If EXBUS is enabled, then these pins will remain TRI-STATE until the SONIC-T becomes a bus master, at which time they will be driven according to the DCR2. If EXBUS is disabled, then these four pins work normally as external ENDEC interface pins.</li> <li>2. Synchronous Termination, STERM: This changes the TXC pin from the External ENDEC interface into a synchronous memory termination input for compatibility with Motorola style processors. This input is only useful when Asynchronous Bus mode is selected (bit 10 below is set to "0") and the Bus Mode is set to Motorola Mode (BMODE = 1). On hardware reset, this pin will be TRI-STATE and will remain that way until the DCR is changed. If EXBUS is enabled, this pin will remain TRI-STATE until the SONIC-T becomes a bus master, at which time it will become the STERM input. If EXBUS is disabled, then this pin works normally as the TXC pin for the external ENDEC interface.</li> <li>3. Asynchronous Bus Retry: Causes <math>\overline{BRT}</math> to be clocked in asynchronously off the falling edge of bus clock. This only applies, however, when the SONIC-T is operating in asynchronous mode (bit 10 below is set to "0"). If EXBUS is not set, XTO (BRT) is sampled synchronously off the rising edge of bus clock.</li> </ol>
14	Must be 0.
13	<p><b>LBR: LATCHED BUS RETRY</b></p> <p>The LBR bit controls the mode of operation of the <math>\overline{BRT}</math> signal (see pin description, Section 2.0). It allows the BUS Retry operation to be latched or unlatched.</p> <p>0: Unlatched mode: The assertion of <math>\overline{BRT}</math> forces the SONIC-T to finish the current DMA operation and get off the bus. The SONIC-T will retry the operation when <math>\overline{BRT}</math> is deasserted.</p> <p>1: Latched mode: The assertion of <math>\overline{BRT}</math> forces the SONIC-T to finish the current DMA operation as above, however, the SONIC-T will not retry until <math>\overline{BRT}</math> is deasserted and the BR bit in the ISR (see Section 6.3.6) has been reset and <math>\overline{BRT}</math> is deasserted. Hence, the mode has been latched on until the BR bit is cleared.</p> <p><b>Note:</b> Unless LBR is set to a "1", <math>\overline{BRT}</math> must remain asserted at least until the SONIC-T has gone idle. See sec. 7.3.6.</p>
12, 11	<p><b>PO1,PO0: PROGRAMMABLE OUTPUTS</b></p> <p>The PO1,PO0 bits individually control the USR1,0 pins respectively when SONIC-T is a bus master (HLDA or <math>\overline{BGACK}</math> is active). When PO1/PO0 are set to a 1 the USR1/USR0 pins are high during bus master operations and when these bits are set to a 0 the USR1/USR0 pins are low during bus master operations.</p>

## 6.0 SONIC-T Registers (Continued)

### 6.3.2 Data Configuration Register (Continued)

Bit	Description															
10	<p><b>SBUS: SYNCHRONOUS BUS MODE</b></p> <p>The SBUS bit is used to select the mode of system bus operation when SONIC-T is a bus master. This bit selects the internal ready line to be either a synchronous or asynchronous input to SONIC-T during block transfer DMA operations.</p> <p>0: Asynchronous mode. <math>\overline{RDY}_i</math> (BMODE = 0) or <math>\overline{DSACK0}_i</math> (BMODE = 1) are respectively internally synchronized at the falling edge of the bus clock (T2 of the DMA cycle). No setup or hold times need to be met with &lt;S2–S5&gt; respect to this edge to guarantee proper bus operation. The minimum memory cycle time is 3 bus clocks.</p> <p>1: Synchronous mode. <math>\overline{RDY}_i</math> (BMODE = 0) and <math>\overline{DSACK0}_i</math> (BMODE = 1) must respectively meet the setup and &lt;S2–S5&gt; hold times with respect to the rising edge of T1 or T2 to guarantee proper bus operation.</p>															
9, 8	<p><b>USR1,0: USER DEFINABLE PINS</b></p> <p>The USR1,0 bits report the level of the USR1,0 signal pins, respectively, after a chip hardware reset. If the USR1,0 signal pins are at a logical 1 (tied to <math>V_{CC}</math>) during a hardware reset the USR1,0 bits are set to a 1. If the USR1,0 pins are at a logical 0 (tied to ground) during a hardware reset the USR1,0 bits are set to a 0. These bits are latched on the rising edge of <math>\overline{RST}</math>. Once set they remain set/reset until the next hardware reset.</p>															
7, 6	<p><b>WC1,0: WAIT STATE CONTROL</b></p> <p>These encoded bits determine the number of additional bus cycles (T2 states) that are added during each DMA cycle.</p> <table border="1"> <thead> <tr> <th>WC1</th> <th>WC0</th> <th>Bus Cycles Added</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>2</td> </tr> <tr> <td>1</td> <td>1</td> <td>3</td> </tr> </tbody> </table>	WC1	WC0	Bus Cycles Added	0	0	0	0	1	1	1	0	2	1	1	3
WC1	WC0	Bus Cycles Added														
0	0	0														
0	1	1														
1	0	2														
1	1	3														
5	<p><b>DW: DATA WIDTH SELECT</b></p> <p>These bits select the data path width for DMA operations.</p> <table border="1"> <thead> <tr> <th>DW</th> <th>Data Width</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>16-bit</td> </tr> <tr> <td>1</td> <td>32-bit</td> </tr> </tbody> </table>	DW	Data Width	0	16-bit	1	32-bit									
DW	Data Width															
0	16-bit															
1	32-bit															
4	<p><b>BMS: BLOCK MODE SELECT FOR DMA</b></p> <p>Determines how data is emptied or filled into the Receive or Transmit FIFO.</p> <p>0: Empty/fill mode: All DMA transfers continue until either the Receive FIFO has emptied or the Transmit FIFO has filled completely.</p> <p>1: Block mode: All DMA transfers continue until the programmed number of bytes RFT0, (RFT1 during reception or TFO, TF1 during transmission) have been transferred. (See note for TFT0, TFT1.)</p>															
3, 2	<p><b>RFT1,RFT0: RECEIVE FIFO THRESHOLD</b></p> <p>These encoded bits determine the number of words (or long words) that are written into the receive FIFO from the MAC unit before a receive DMA request occurs. (See Section 3.5.)</p> <table border="1"> <thead> <tr> <th>LB1</th> <th>LB0</th> <th>Function</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>2 words or 1 long word (4 bytes)</td> </tr> <tr> <td>0</td> <td>1</td> <td>4 words or 2 long words (8 bytes)</td> </tr> <tr> <td>1</td> <td>0</td> <td>8 words or 4 long words (16 bytes)</td> </tr> <tr> <td>1</td> <td>1</td> <td>12 words or 6 long words (24 bytes)</td> </tr> </tbody> </table> <p><b>Note:</b> In block mode (BMS bit = 1), the receive FIFO threshold sets the number of words (or long words) written to memory during a receive DMA block cycle.</p>	LB1	LB0	Function	0	0	2 words or 1 long word (4 bytes)	0	1	4 words or 2 long words (8 bytes)	1	0	8 words or 4 long words (16 bytes)	1	1	12 words or 6 long words (24 bytes)
LB1	LB0	Function														
0	0	2 words or 1 long word (4 bytes)														
0	1	4 words or 2 long words (8 bytes)														
1	0	8 words or 4 long words (16 bytes)														
1	1	12 words or 6 long words (24 bytes)														
1, 0	<p><b>TFT1,TFT0: TRANSMIT FIFO THRESHOLD</b></p> <p>These encoded bits determine the minimum number of words (or long words) the DMA section maintains in the transmit FIFO. A bus request occurs when the number of words drops below the transmit FIFO threshold. (See Section 3.5.)</p> <table border="1"> <thead> <tr> <th>LB1</th> <th>LB0</th> <th>Function</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>4 words or 2 long words (8 bytes)</td> </tr> <tr> <td>0</td> <td>1</td> <td>18 words or 4 long words (16 bytes)</td> </tr> <tr> <td>1</td> <td>0</td> <td>12 words or 6 long words (24 bytes)</td> </tr> <tr> <td>1</td> <td>1</td> <td>14 words or 7 long words (28 bytes)</td> </tr> </tbody> </table> <p><b>Note:</b> In block mode (BMS = 1), the number of bytes the SONIC-T reads in a single DMA burst equals the transmit FIFO threshold value. If the number of words or long words needed to fill the FIFO is less than the threshold value, then only the number of reads required to fill the FIFO in a single DMA burst will be made. Typically, with the FIFO threshold value set to 12 or 14 words, the number of memory reads needed is less than the FIFO threshold value.</p>	LB1	LB0	Function	0	0	4 words or 2 long words (8 bytes)	0	1	18 words or 4 long words (16 bytes)	1	0	12 words or 6 long words (24 bytes)	1	1	14 words or 7 long words (28 bytes)
LB1	LB0	Function														
0	0	4 words or 2 long words (8 bytes)														
0	1	18 words or 4 long words (16 bytes)														
1	0	12 words or 6 long words (24 bytes)														
1	1	14 words or 7 long words (28 bytes)														

## 6.0 SONIC-T Registers (Continued)

### 6.3.3 Receive Control Register

(RA<5:0> = 2h)

This register is used to filter incoming packets and provide status information of accepted packets (Figure 6-6). Setting any of bits 15–11 to a “1” enables the corresponding receive filter. If none of these bits are set, only packets which match the CAM Address registers are accepted. Bits 10 and 9 control the loopback operations.

After reception, bits 8–0 indicate status information about the accepted packet and are set to “1” when the corresponding condition is true. If the packet is accepted, all bits in the RCR are written into the RXpkt.status field. Bits 8–6 and 3–0 are cleared at the reception of the next packet.

This register is unaffected by a software reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR	RNT	BRD	PRO	AMC	LB1	LB0	MC	BC	LPKT	CRS	COL	CRCR	FAER	LBK	PRX
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r	r	r	r	r	r	r	r	r

r = read only, r/w = read/write

**FIGURE 6-6. Receive Control Register**

Field	Meaning
ERR	ACCEPT PACKET WITH ERRORS
RNT	ACCEPT RUNT PACKETS
BRD	ACCEPT BROADCAST PACKETS
PRO	PHYSICAL PROMISCUOUS PACKETS
AMC	ACCEPT ALL MULTICAST PACKETS
LB0, LB1	LOOPBACK CONTROL
MC	MULTICAST PACKET RECEIVED
BC	BROADCAST PACKET RECEIVED
LPKT	LAST PACKET IN RBA
CRS	CARRIER SENSE ACTIVITY
COL	COLLISION ACTIVITY
CRCR	CRC ERROR
FAER	FRAME ALIGNMENT ERROR
LBK	LOOPBACK PACKET RECEIVED
PRX	PACKET RECEIVED OK

Bit	Description
15	<b>ERR: ACCEPT PACKET WITH CRC ERRORS OR COLLISIONS</b> 0: Reject all packets with CRC errors or when a collision occurs. 1: Accept packets with CRC errors and ignore collisions.
14	<b>RNT: ACCEPT RUNT PACKETS</b> 0: Normal address match mode. 1: Accept runt packets (packets less than 64 bytes in length). <i>Note:</i> A hardware reset clears this bit.
13	<b>BRD: ACCEPT BROADCAST PACKETS</b> 0: Normal address match mode. 1: Accept broadcast packets (packets with addresses that match the CAM are also accepted). <i>Note:</i> This bit is cleared upon hardware reset.
12	<b>PRO: PHYSICAL PROMISCUOUS MODE</b> Enable all Physical Address packets to be accepted. 0: normal address match mode. 1: promiscuous mode.
11	<b>AMC: ACCEPT ALL MULTICAST PACKETS</b> 0: normal address match mode. 1: enables all multicast packets to be accepted. Broadcast packets are also accepted regardless of the BRD bit. (Broadcast packets are a subset of multicast packets.)

## 6.0 SONIC-T Registers (Continued)

### 6.3.3 Receive Control Register (Continued)

Bit	Description															
10, 9	<p><b>LB1, LB0: LOOPBACK CONTROL</b>            These encoded bits control loopback operations for MAC loopback, ENDEC loopback and Transceiver lookback. For proper loopback operation, the CAM Address registers and Receive Control register must be initialized to accept the Destination address of the loopback packet (see Section 3.8).</p> <table border="1"> <thead> <tr> <th>LB1</th> <th>LB0</th> <th>Function</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>no loopback, normal operation</td> </tr> <tr> <td>0</td> <td>1</td> <td>MAC loopback</td> </tr> <tr> <td>1</td> <td>0</td> <td>ENDEC loopback</td> </tr> <tr> <td>1</td> <td>1</td> <td>Transceiver loopback (Bit #6 of the DCR2 must also be set to 1)</td> </tr> </tbody> </table>	LB1	LB0	Function	0	0	no loopback, normal operation	0	1	MAC loopback	1	0	ENDEC loopback	1	1	Transceiver loopback (Bit #6 of the DCR2 must also be set to 1)
LB1	LB0	Function														
0	0	no loopback, normal operation														
0	1	MAC loopback														
1	0	ENDEC loopback														
1	1	Transceiver loopback (Bit #6 of the DCR2 must also be set to 1)														
8	<p><b>MC: MULTICAST PACKET RECEIVED</b>            This bit is set when a packet is received with a Multicast Address.</p>															
7	<p><b>BC: BROADCAST PACKET RECEIVED</b>            This bit is set when a packet is received with a Broadcast Address.</p>															
6	<p><b>LPKT: LAST PACKET IN RBA</b>            This bit is set when the last packet is buffered into a Receive Buffer Area (RBA). The SONIC-T detects this condition when its Remaining Buffer Word Count (RBWC0,1) register is less than or equal to the End Of Buffer Count (EOBC) register. (See Section 5.4.2.)</p>															
5	<p><b>CRS: CARRIER SENSE ACTIVITY</b>            Set when CRS is active. Indicates the presence of network activity.</p>															
4	<p><b>COL: COLLISION ACTIVITY</b>            Indicates that the packet received had a collision occur during reception.</p>															
3	<p><b>CRCR: CRC ERROR</b>            Indicates the packet contains a CRC error. If the packet also contains a Frame Alignment error, FAER will be set instead (see below).</p>															
2	<p><b>FAER: FRAME ALIGNMENT ERROR</b>            Indicates that the incoming packet was not correctly framed on an 8-bit boundary. Note: if no CRC errors have occurred, this bit is not set (i.e., this bit is only set when both a frame alignment and CRC errors occur).</p>															
1	<p><b>LBK: LOOPBACK PACKET RECEIVED</b>            Indicates that the SONIC-T has successfully received a loopback packet.</p>															
0	<p><b>PRX: PACKET RECEIVED OK</b>            Indicates that a packet has been received without CRC, frame alignment, length (run packet) errors or collisions.</p>															

## 6.0 SONIC-T Registers (Continued)

### 6.3.4 Transmit Control Register

(RA <5:0> = 3h)

This register is used to program the SONIC-T's transmit actions and provide status information after a packet has been transmitted (*Figure 6-7*). At the beginning of transmission, bits 15, 14, 13 and 12 from the TXpkt.config field are loaded into the TCR to configure the various transmit modes (see Section 5.5.1.1). When the transmission ends, bits 10–0 indicate status information and are set to a "1" when the corresponding condition is true. These bits, along with the number of collisions information, are written into the TXpkt.status field at the end of transmission (see Section 5.5.1.2). Bits 9 and 5 are cleared after the TXpkt.status field has been written. Bits 10, 7, 6, and 1 are cleared at the commencement of the next transmission while bit 8 is set at this time.

A hardware reset sets bits 8 and 0 to a "1", and bit 1 to a "0". ∴ This register is unaffected by a software reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PINT	POWC	CRCI	EXDIS	0	EXD	DEF	NCRS	CRSL	EXC	OWC	0	PMB	FU	BCM	PTX
r/w	r/w	r/w	r/w	r	r	r	r	r	r	r	r	r	r	r	r

r = read only, r/w = read/write

**FIGURE 6-7. Transmit Control Register**

Field	Meaning
PINT	PROGRAMMABLE INTERRUPT
POWC	PROGRAMMED OUT OF WINDOW COLLISION TIMER
CRCI	CRC INHIBIT
EXDIS	DISABLE EXCESSIVE DEFERRAL TIMER
EXD	EXCESSIVE DEFERRAL
DEF	DEFERRED TRANSMISSION
NCRS	NO CRS
CRSL	CRS LOST
EXC	EXCESSIVE COLLISIONS
OWC	OUT OF WINDOW COLLISION
PMB	PACKET MONITORED BAD
FU	FIFO UNDERRUN
BCM	BYTE COUNT MISMATCH
PTX	PACKET TRANSMITTED OK

Bit	Description
15	<p><b>PINTR: PROGRAMMABLE INTERRUPT</b></p> <p>This bit allows transmit interrupts to be generated under software control. The SONIC-T will issue an interrupt (PINT in the Interrupt Status Register) immediately after reading a TDA and detecting that PINT is set in the TXpkt.config field.</p> <p><b>Note:</b> In order for PINT to operate properly, it must be set and reset in the TXpkt.config field by alternating TDAs. This is necessary because after PINT has been issued in the ISR, PINT in the Transmit Control Register must be cleared before it is set again in order to have the interrupt issued for another packet. The only effective way to do this is to set PINT to a 1 no more often than every other packet.</p>
14	<p><b>POWC: PROGRAM "OUT OF WINDOW COLLISION" TIMER</b></p> <p>This bit programs when the out of window collision timer begins.</p> <p>0: timer begins after the Start of Frame Delimiter (SFD).</p> <p>1: timer begins after the first bit of preamble.</p>
13	<p><b>CRCI: CRC INHIBIT</b></p> <p>0: transmit packet with 4-byte FCS field</p> <p>1: transmit packet without 4-byte FCS field</p>
12	<p><b>EXDIS: DISABLE EXCESSIVE DEFERRAL TIMER:</b></p> <p>0: excessive deferral timer enabled</p> <p>1: excessive deferral timer disabled</p>
11	Must be 0.
10	<p><b>EXD: EXCESSIVE DEFERRAL</b></p> <p>Indicates that the SONIC-T has been deferring for 3.2 ms. The transmission is aborted if the excessive deferral timer is enabled (i.e., EXDIS is reset). This bit can only be set if the excessive deferral timer is enabled.</p>



## 6.0 SONIC-T Registers (Continued)

### 6.3.4 Transmit Control Register (Continued)

Bit	Description
9	<p><b>DEF: DEFERRED TRANSMISSION</b></p> <p>Indicates that the SONIC-T has deferred its transmission during the first attempt. If subsequent collisions occur, this bit is reset. This bit is cleared after the TXpkt.status field is written in the TDA.</p>
8	<p><b>NCRS: NO CRS</b></p> <p>Indicates that Carrier Sense (CRS) was not present during transmission. CRS is monitored from the beginning of the Start of Frame Delimiter to the last byte transmitted. The transmission will not be aborted. This bit is set at the start of preamble and is reset if CRS is detected. Hence, if CRS is never detected throughout the entire transmission of the packet, this bit will remain set.</p> <p><b>Note:</b> NCRS will always remain set in MAC loopback as long as there is no activity on the RX ±.</p>
7	<p><b>CRSL: CRS LOST</b></p> <p>Indicates that CRS has gone low or has not been present during transmission. CRS is monitored from the beginning of the Start of Frame Delimiter to the last byte transmitted. The transmission will not be aborted.</p> <p><b>Note:</b> if CRS was never present, both NCRS and CRSL will be set simultaneously. Also, CRSL will always be set in MAC loopback.</p>
6	<p><b>EXC: EXCESSIVE COLLISIONS</b></p> <p>Indicates that 16 collisions have occurred. The transmission is aborted.</p>
5	<p><b>OWC: OUT OF WINDOW COLLISION</b></p> <p>Indicates that an illegal collision has occurred after 51.2 μs (one slot time) from either the first bit of preamble or from SFD depending upon the POWC bit. The transmission backs off as in a normal transmission. This bit is cleared after the TXpkt.status field is written in the TDA.</p>
4	Must be 0.
3	<p><b>PMB: PACKET MONITORED BAD</b></p> <p>This bit is set, if after the receive unit has monitored the transmitted packet, the CRC has been calculated as invalid as a result of a Frame Alignment error, or the Source Address does not match any of the CAM address registers.</p> <p><b>Note 1:</b> The SONIC-T's CRC checker is active during transmission.</p> <p><b>Note 2:</b> If CRC has been inhibited for transmissions (CRCI is set), this bit will always be low. This is true regardless of Frame Alignment or Source Address mismatch errors.</p> <p><b>Note 3:</b> If a Receive FIFO overrun has occurred, the transmitted packet is not monitored completely. Thus, if PMB is set along with the RFO bit in the ISR, then PMB has no meaning. The packet must be completely received before PMB has meaning.</p> <p><b>Note 4:</b> This bit is always zero in loopback mode. (True for all three types of loopback mode.)</p>
2	<p><b>FU: FIFO UNDERRUN</b></p> <p>Indicates that the SONIC-T has not been able to access the bus before the FIFO has emptied. This condition occurs from excessive bus latency and/or slow bus clock. The transmission is aborted. (See Section 3.5.2.)</p>
1	<p><b>BCM: BYTE COUNT MISMATCH</b></p> <p>This bit is set when the SONIC-T detects that the TXpkt.pkt_size field is not equal to the sum of the TXpkt.frag_size field(s). Transmission is aborted. This bit will also be set when Excessive Collisions (bit 6 of the transmit control register) occur during transmission.</p>
0	<p><b>PTX: PACKET TRANSMITTED OK</b></p> <p>Indicates that a packet has been transmitted without the following errors:</p> <ul style="list-style-type: none"> <li>—Excessive Collisions (EXC)</li> <li>—Excessive Deferral (EXD)</li> <li>—FIFO Underrun (FU)</li> <li>—Byte Count Mismatch (BCM)</li> </ul>

## 6.0 SONIC-T Registers (Continued)

### 6.3.5 Interrupt Mask Register

(RA<5:0> = 4h)

This register masks the interrupts that can be generated from the ISR (Figure 6-8). Writing a "1" to the bit enables the corresponding interrupt. During a hardware reset, all mask bits are cleared.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	BREN	HBLEN	LCDEN	PINTEN	PRXEN	PTXEN	TXEREN	TCEN	RDEEN	RBEEN	RBAEEN	CRCEN	FAEEN	MPEN	RFOEN
	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

r/w = read/write

FIGURE 6-8. Interrupt Mask Register

Field	Meaning
BREN	BUS RETRY OCCURRED ENABLE
HBLEN	HEARTBEAT LOST ENABLE
LCDEN	LOAD CAM DONE INTERRUPT ENABLE
PINTEN	PROGRAMMABLE INTERRUPT ENABLE
PRXEN	PACKET RECEIVED ENABLE
PTXEN	PACKET TRANSMITTED OK ENABLE
TXEREN	TRANSMIT ERROR ENABLE
TCEN	TIMER COMPLETE ENABLE
RDEEN	RECEIVE DESCRIPTORS ENABLE
RBEEN	RECEIVE BUFFERS EXHAUSTED ENABLE
RBAEEN	RECEIVE BUFFER AREA EXCEEDED ENABLE
CRCEN	CRC TALLY COUNTER WARNING ENABLE
FAEEN	FAE TALLY COUNTER WARNING ENABLE
MPEN	MP TALLY COUNTER WARNING ENABLE
RFOEN	RECEIVE FIFO OVERRUN ENABLE

Bit	Description
15	<b>Must be 0</b>
14	<b>BREN: BUS RETRY OCCURRED enable:</b> 0: disable 1: enables interrupts when a Bus Retry operation is requested.
13	<b>HBLEN: HEARTBEAT LOST enable:</b> 0: disable 1: enables interrupts when a heartbeat lost condition occurs.
12	<b>LCDEN: LOAD CAM DONE INTERRUPT enable:</b> 0: disable 1: enables interrupts when the Load CAM command has finished.
11	<b>PINTEN: PROGRAMMABLE INTERRUPT enable:</b> 0: disable 1: enables programmable interrupts to occur when the PINT bit the TXpkt.config field is set to a "1".
10	<b>PRXEN: PACKET RECEIVED enable:</b> 0: disable 1: enables interrupts for packets accepted.
9	<b>PTXEN: PACKET TRANSMITTED OK enable:</b> 0: disable 1: enables interrupts for transmit completions.
8	<b>TXEREN: TRANSMIT ERROR enable:</b> 0: disable 1: enables interrupts for packets transmitted with error.

## 6.0 SONIC-T Registers (Continued)

### 6.3.5 Interrupt Mask Register (Continued)

Bit	Description
7	<b>TCEN: GENERAL PURPOSE TIMER COMPLETE enable:</b> 0: disable 1: enables interrupts when the general purpose timer has rolled over from 0000 0000h to FFFF FFFFh.
6	<b>RDEEN: RECEIVE DESCRIPTORS EXHAUSTED enable:</b> 0: disable 1: enables interrupts when all receive descriptors in the RDA have been exhausted.
5	<b>RBEEN: RECEIVE BUFFERS EXHAUSTED enable:</b> 0: disable 1: enables interrupts when all resource descriptors in the RRA have been exhausted.
4	<b>RBAEEN: RECEIVE BUFFER AREA EXCEEDED enable:</b> 0: disable 1: enables interrupts when the SONIC-T attempts to buffer data beyond the end of the Receive Buffer Area.
3	<b>CRGEN: CRC TALLY COUNTER WARNING enable:</b> 0: disable 1: enables interrupts when the CRC tally counter has rolled over from FFFFh to 0000h.
2	<b>FAEEN: FRAME ALIGNMENT ERROR (FAE) TALLY COUNTER WARNING enable:</b> 0: disable 1: enables interrupts when the FAE tally counter rolled over from FFFFh to 0000h.
1	<b>MPEN: MISSED PACKET (MP) TALLY COUNTER WARNING enable:</b> 0: disable 1: enables interrupts when the MP tally counter has rolled over from FFFFh to 0000h.
0	<b>RFOEN: RECEIVE FIFO OVERRUN enable:</b> 0: disable 1: enables interrupts when the receive FIFO has overrun.

## 6.0 SONIC-T Registers (Continued)

### 6.3.6 Interrupt Status Register

(RA<5:0> = 5h)

This register (*Figure 6-9*) indicates the source of an interrupt when the INT pin goes active. Enabling the corresponding bits in the IMR allows bits in this register to produce an interrupt. When an interrupt is active, one or more bits in this register are set to a "1". A bit is cleared by writing "1" to it. Writing a "0" to any bit has no effect.

This register is cleared by a hardware reset and unaffected by a software reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	BR	HBL	LCD	PINT	PKTRX	PTDN	TXER	TC	RDE	RBE	RBAE	CRC	FAE	MP	RFO
	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

r/w=read/write

**FIGURE 6-9. Interrupt Status Register**

Field	Meaning
BR	BUS RETRY OCCURRED
HBL	CD HEARTBEAT LOST
LCD	LOAD CAM DONE
PINT	PROGRAMMABLE INTERRUPT
PKTRX	PACKET RECEIVED
TXDN	TRANSMISSION DONE
TXER	TRANSMIT ERROR
TC	TIMER COMPLETE
RDE	RECEIVE DISCRIPTORS EXHAUSTED
RBE	RECEIVE BUFFERS EXHAUSTED
RBAE	RECEIVE BUFFER AREA EXCEEDED
CRC	CRC TALLY COUNTER ROLLOVER
FAE	FRAME ALIGNMENT ERROR
MP	MISSED PACKET COUNTER ROLLOVER
RFO	RECEIVE FIFO OVERRUN

Bit	Description
15	<b>Must be 0</b>
14	<p><b>BR: BUS RETRY OCCURRED</b></p> <p>Indicates that a Bus Retry (BRT) operation has occurred. In Latched Bus Retry mode (LBR in the DCR), BR will only be set when the SONIC-T is a bus master. Before the SONIC-T will continue any DMA operations, BR must be cleared. In Unlatched mode, the BR bit should be cleared also, but the SONIC-T will not wait for BR to be cleared before requesting the bus again and continuing its DMA operations. (See Sections 6.3.2 and 7.3.6 for more information on Bus Retry.)</p>
13	<p><b>HBL: CD HEARTBEAT LOST</b></p> <p>If the transceiver fails to provide a collision pulse (heart beat) during the first 6.4 μs of the Interframe Gap after transmission, this bit is set.</p>
12	<p><b>LCD: LOAD CAM DONE</b></p> <p>Indicates that the Load CAM command has finished writing to all programmed locations in the CAM. (See Section 6.1.1.)</p>
11	<p><b>PINT: PROGRAMMED INTERRUPT</b></p> <p>Indicates that upon reading the TXpkt.config field, the SONIC-T has detected the PINT bit to be set. (See Section 6.3.4.)</p>
10	<p><b>PKTRX: PACKET RECEIVED</b></p> <p>Indicates that a packet has been received and been buffered to memory. This bit is set after the RXpkt.seq_no field is written to memory.</p>
9	<p><b>TXDN: TRANSMISSION DONE</b></p> <p>Indicates that either (1) there are no remaining packets to be transmitted in the Transmit Descriptor Area (i.e., the EOL bit has been detected as a "1"), (2) the Halt Transmit command has been given (HTX bit in CR is set to a "1"), or (3) a transmit abort condition has occurred. This condition occurs when any of following bits in the TCR are set: BCM, EXC, FU, or EXD. This bit is set after the TXpkt.status field has been written to.</p>

## 6.0 SONIC-T Registers (Continued)

### 6.3.6 Interrupt Status Register (Continued)

Bit	Description
8	<p><b>TXER: TRANSMIT ERROR</b></p> <p>Indicates that a packet has been transmitted with at least one of the following errors.</p> <ul style="list-style-type: none"> <li>—Byte count mismatch (BCM)</li> <li>—Excessive collisions (EXC)</li> <li>—FIFO underrun (FU)</li> <li>—Excessive deferral (EXD)</li> </ul> <p>The TXpkt.status field reveals the cause of the error(s).</p>
7	<p><b>TC: GENERAL PURPOSE TIMER COMPLETE</b></p> <p>Indicates that the timer has rolled over from 0000 0000h to FFFF FFFFh. (See Section 6.3.12.)</p>
6	<p><b>RDE: RECEIVE DESCRIPTORS EXHAUSTED</b></p> <p>Indicates that all receive packet descriptors in the RDA have been exhausted. This bit is set when the SONIC-T detects EOL = 1. (See Section 5.4.7.)</p>
5	<p><b>RBE: RECEIVE BUFFER EXHAUSTED</b></p> <p>Indicates that the SONIC-T has detected the Resource Read Pointer (RRP) is equal to the Resource Write Pointer (RWP). This bit is set after the last field is read from the resource area. (See Section 5.4.7.)</p> <p><b>Note 1:</b> This bit will be set as the SONIC-T finishes using the second to last receive buffer and reads the last RRA descriptor. This gives the system an early warning of impending no resources.</p> <p><b>Note 2:</b> The SONIC-T will stop reception of packets when the last RBA has been used and will not continue reception until additional receive buffers have been added (i.e., RWP is incremented beyond RRP) and this bit has been reset.</p> <p><b>Note 3:</b> If additional buffers have been added, resetting this bit causes the SONIC-T to read the next resource descriptor pointed to by the RRP in the Receive Resource Area. Note that resetting this bit under this condition is similar to issuing the Read RRA command (setting the RRRR bit in the Command Register). This bit should never be reset until after the additional resources have been added to the RRA.</p>
4	<p><b>RBAE: RECEIVE BUFFER AREA EXCEEDED</b></p> <p>Indicates that during reception, the SONIC-T has reached the end of the Receive Buffer Area. Reception is aborted and the SONIC-T fetches the next available resource descriptors in the RRA. The buffer space is not re-used and an RDA is not set up for the truncated packet (see Section 5.4.7).</p>
3	<p><b>CRC: CRC TALLY COUNTER ROLLOVER</b></p> <p>Indicates that the tally counter has rolled over from FFFFh to 0000h. (See Section 6.3.11.)</p>
2	<p><b>FAE: FRAME ALIGNMENT ERROR (FAE) TALLY COUNTER ROLLOVER</b></p> <p>Indicates that the FAE tally counter has rolled over from FFFFh to 0000h. (See Section 6.3.11.)</p>
1	<p><b>MP: MISSED PACKET (MP) COUNTER ROLLOVER</b></p> <p>Indicates that the MP tally counter has rolled over from FFFFh to 0000h. (See Section 6.3.11.)</p>
0	<p><b>RFO: RECEIVE FIFO OVERRUN</b></p> <p>Indicates that the SONIC-T has been unable to access the bus before the receive FIFO has filled from the network. This condition is due to excessively long bus latency and/or slow bus clock. Note that FIFO underruns are indicated in the TCR. (See Section 3.5.1.)</p>

## 6.0 SONIC-T Registers (Continued)

### 6.3.7 Data Configuration Register 2

(RA<5:0> = 3Fh)

This register (Figure 6-10) is for enabling the extended bus interface options.

A hardware reset will set all bits in this register to "0" except for the Extended Programmable Outputs which are unknown until written to. A software reset will not affect any bits in this register. This register should only be written to when the SONIC-T is in software reset (the  $\overline{\text{RST}}$  bit in the Command Register is set).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXPO3	EXPO2	EXPO1	EXPO0	HD	0	JD	AUTO	0	XWRAP	0	PH	0	PCM	PCNM	RJCM
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

**FIGURE 6-10. Data Configuration Register 2**

Field	Meaning
EXPO3-0	EXTENDED PROGRAMMABLE OUTPUTS
HD	HEART BEAT DISABLE
JD	TPI JABBER TIMER DISABLE
AUTO	AUI/TPI AUTO SELECTION
XWRAP	TPI TRANSCEIVER LOOPBACK
PH	PROGRAM HOLD
PCM	PACKET COMPRESS WHEN MATCHED
PCNM	PACKET COMPRESS WHEN NOT MATCHED
RJCM	REJECT ON CAM MATCH

Bit	Description
15-12	<p><b>EXPO &lt;3:0&gt; EXTENDED PROGRAMMABLE OUTPUTS</b></p> <p>These bits program the level of the Extended User outputs (EXUSR &lt;3:0&gt;) when the SONIC-T is a bus master. Writing a "1" to any of these bits programs a high level to the corresponding output. Writing a "0" to any of these bits programs a low level to the corresponding output. EXUSR &lt;3:0&gt; are similar to USR &lt;1:0&gt; except that EXUSR &lt;3:0&gt; are only available when the Extended Bus mode is selected (bit 15 in the DCR is set to "1", see Section 6.3.2).</p>
11	<p><b>HD: HEART BEAT DISABLE</b></p> <p>This bit allows the SONIC-T to ignore the heart beat signal.</p> <p>0: Enable heart beat. 1: Disable heart beat.</p>
9	<p><b>JD: TPI JABBER TIMER DISABLE</b></p> <p>This bit allows the user to turn on/off the jabber timer.</p> <p>0: Enable the jabber timer. 1: Disable the jabber timer.</p>
8	<p><b>AUTO: AUI/TPI AUTO SELECTION</b></p> <p>This bit allows the SONIC-T to check for a good link on the TPI and AUI. SONIC-T will first look for a good link on the TPI. If there is no good link on the TPI, SONIC-T will automatically select the AUI. If this bit is enable, the AUI/TPI pin will be ignored.</p> <p>0: Disable AUI/TPI auto selection. 1: Enable AUI/TPI auto selection.</p>
7	Must be zero.
6	<p><b>XWRAP: TPI TRANSCEIVER LOOPBACK</b></p> <p>This bit controls the loopback operation for the TPI transceiver. For proper operation, the CAM Address Registers and Receiver Control Register must be initialized to accept the destination address of the loopback packet (refer to loopback procedure). Also, both bits 9 and 10 of the RCR must be set to 1.</p> <p>0: Disable TPI transceiver loopback. 1: Enable TPI transceiver loopback.</p>
5	Must be zero.
4	<p><b>PH: PROGRAM HOLD</b></p> <p>When this bit is set to "0", the HOLD request output is asserted/deasserted from the falling edge of bus clock. If this bit is set to "1", HOLD will be asserted/deasserted 1/2 clock later on the rising edge of bus clock.</p>
3	Must be zero.

## 6.0 SONIC-T Registers (Continued)

### 6.3.7 Data Configuration Register 2 (Continued)

Bit	Description
2	<p><b>PCM: PACKET COMPRESS WHEN MATCHED</b></p> <p>When this bit is set to a "1" (and the PCNM bit is reset to a "0"), the <math>\overline{\text{PCOMP}}</math> output will be asserted if the destination address of the packet being received matches one of the entries in the CAM (Content Addressable Memory). This bit, along with PCNM, is used with the Management Bus of the DP83950, Repeater Interface Controller (RIC). See the DP83950 datasheet for more details on the RIC Management Bus. This mode is also called the Managed Bridge Mode.</p> <p><b>Note 1:</b> Setting PCNM and PCM to "1" at the same time is not allowed.</p> <p><b>Note 2:</b> If PCNM and PCM are both "0", the <math>\overline{\text{PCOMP}}</math> output will remain TRI-STATE until PCNM or PCM are changed.</p>
1	<p><b>PCNM: COMPRESS WHEN NOT MATCHED</b></p> <p>When this bit is set to a "1" (and the PCM bit is set to "0"), the <math>\overline{\text{PCOMP}}</math> output will be asserted if the destination address of the packet does not match one of the entries in the CAM. See the PCM bit above. This mode is also called the Managed Hub Mode.</p> <p><b>Note:</b> <math>\overline{\text{PCOMP}}</math> will not be asserted if the destination address is a broadcast address. This is true regardless of the state of the BRD bit in the Receive Control Register.</p>
0	<p><b>RJCM: REJECT ON CAM MATCH</b></p> <p>When this bit is set to "1", the SONIC-T will reject a packet on a CAM match. Setting RJCM to "0" causes the SONIC-T to operate normally by accepting packets on a CAM match. Setting this mode is useful for a small bridge with a limited number of nodes attached to it. RJCM only affects the CAM, though. Setting RJCM will not invert the function of the BRD, PRO or AMC bits (to accept broadcast, all physical or multicast packets respectively) in the Receive Control Register (see Section 6.3.3). This means, for example, that it is not possible to set RJCM and BRD to reject all broadcast packets. If RJCM and BRD are set at the same time, however, all broadcast packets will be accepted, but any packets that have a destination address that matches an address in the CAM will be rejected.</p>

## 6.0 SONIC-T Registers (Continued)

### 6.3.8 Transmit Registers

The transmit registers described in this section are part of the User Register set. The UTDA and CTDA must be initialized prior to issuing the transmit command (setting the TXP bit) in the Command register.

**Upper Transmit Descriptor Address Register (UTDA):** This register contains the upper address bits ( $A<31:16>$ ) for accessing the transmit descriptor area (TDA) and is concatenated with the contents of the CTDA when the SONIC-T accesses the TDA in system memory. The TDA can be as large as 32k words or 16k long words and can be located anywhere in system memory. This register is unaffected by a hardware or software reset.

**Current Transmit Descriptor Address Register (CTDA):** The 16-bit CTDA register contains the lower address bits ( $A<15:1>$ ) of the 32-bit transmit descriptor address. During initialization this register must be programmed with the lower address bits of the transmit descriptor. The SONIC-T concatenates the contents of this register with the contents of the UTDA to point to the transmit descriptor. For 32-bit memory systems, bit 1, corresponding to address signal A1, must be set to "0" for alignment to long-word boundaries. Bit 0 of this register is the End of List (EOL) bit and is used to denote the end of the list. This register is unaffected by a hardware or software reset.

### 6.3.9 Receive Registers

The receive registers described in this section are part of the User Register set. A software reset has no effect on these registers and a hardware reset only affects the EOBC and RSC registers. The receive registers must be initialized prior to issuing the receive command (setting the RXEN bit) in the Command register.

**Upper Receive Descriptor Address Register (URDA):** This register contains the upper address bits ( $A<31:16>$ ) for accessing the receive descriptor area (RDA) and is concatenated with the contents of the CRDA when the SONIC-T accesses the RDA in system memory. The RDA can be as large as 32k words or 16k long words and can be located anywhere in system memory. This register is unaffected by a hardware or software reset.

**Current Receive Descriptor Address Register (CRDA):** The CRDA is a 16-bit read/write register used to locate the received packet descriptor block within the RDA. It contains the lower address bits ( $A<15:1>$ ). The SONIC-T concatenates the contents of the CRDA with the contents of the URDA to form the complete 32-bit address. The resulting 32-bit address points to the first field of the descriptor block. For 32-bit memory systems, bit 1, corresponding to address signal A1, must be set to "0" for alignment to long-word boundaries. Bit 0 of this register is the End of List (EOL) bit and is used to denote the end of the list. This register is unaffected by a hardware or software reset.

**End of Buffer Word Count Register (EOBC):** The SONIC-T uses the contents of this register to determine where to place the next packet. At the end of packet reception, the SONIC-T compares the contents of the EOBC register with the contents of the Remaining Buffer Word Count registers ( $RBWC0,1$ ) to determine whether: (1) to place the next packet in the same RBA or (2) to place the next packet in another RBA. If the EOBC is less than or equal to the remaining number of words in the RBA after a packet is received (i.e.,  $EOBC \leq RBWC0,1$ ), the SONIC-T buffers the next packet in the same RBA. If the EOBC is greater than

the remaining number of words in the RBA after the packet is received (i.e.,  $EOBC > RBWC0,1$ ), the Last Packet in RBA bit, LPKT in the Receive Control Register, Section 6.3.3, is set and the SONIC-T fetches the next resource descriptor. Hence, the next packet received will be buffered in a new RBA. A hardware reset sets this register to 02FH (760 words or 1520 bytes). See Sections 5.4.2 and 5.4.4.4 for more information about using EOBC.

**Upper Receive Resource Address Register (URRA):** The URRA is a 16-bit read/write register. It is programmed with the base address of the receive resource area (RRA). This 16-bit upper address value ( $A<31:16>$ ) locates the receive resource area in system memory. SONIC-T uses the URRA register when accessing the receive descriptors within the RRA by concatenating the lower address value from one of four receive resource registers (RSA, REA, RWP, or RRP).

**Resource Start Address Register (RSA):** The RSA is a 15-bit read/write register. The LSB is not used and always reads back as a 0. The RSA is programmed with the lower 15 bits ( $A<15:1>$ ) of the starting address of the receive resource area. SONIC-T concatenates the contents of this register with the contents of the URRA to form the complete 32-bit address.

**Resource End Address Register (REA):** The REA is a 15-bit read/write register. The LSB is not used and always reads back as a 0. The REA is programmed with the lower 15 bits ( $A<15:1>$ ) of the ending address of the receive resource area. SONIC-T concatenates the contents of this register with the contents of the URRA to form the complete 32-bit address.

**Resource Read Pointer Register (RRP):** The RRP is a 15-bit read/write register. The LSB is not used and always reads back as a 0. The RRP is programmed with the lower 15-bit address ( $A<15:1>$ ) of the first field of the next descriptor the SONIC-T will read. SONIC-T concatenates the contents of this register with the contents of the URRA to form the complete 32-bit address.

**Resource Write Pointer Register (RWP):** The RWP is a 15-bit read/write register. The LSB is not used and always reads back as a 0. The RWP is programmed with the lower 15-bit address ( $A<15:1>$ ) of the next available location the system can add a descriptor. SONIC-T concatenates the contents of this register with the contents of the URRA to form the complete 32-bit address. In 32-bit mode, bit 1, corresponding to address signal A1, must be zero to insure the proper equality comparison between this register and the RRP register.

**Receive Sequence Counter Register (RSC):** This is a 16-bit read/write register containing two fields (Figure 6-11). The SONIC-T uses this register to provide status information on the number of packets within a RBA and the number of RBAs. The RSC register contains two 8-bit (modulo 256) counters. After each packet is received the packet sequence number is incremented. The SONIC-T maintains a single sequence number for each RBA. When the SONIC-T uses the next RBA, the packet sequence number is reset to zero and the RBA sequence number is incremented. This register is reset to 0 by a hardware reset or by writing zero to it. A software reset has no affect.

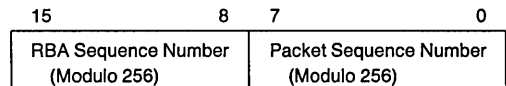


FIGURE 6-11. Receive Sequence Counter Register



## 6.0 SONIC-T Registers (Continued)

### 6.3.10 CAM Registers

The CAM registers described in this section are part of the User Register set. They are used to program the Content Addressable Memory (CAM) entries that provide address filtering of packets. These registers, except for the CAM Enable register, are unaffected by a hardware or software reset.

**CAM Entry Pointer Register (CEP):** The CEP is a 4-bit register used by SONIC-T to select one of the sixteen CAM entries. SONIC-T uses the least significant 4-bits of this register. The value of 0h points to the first CAM entry and the value of Fh points to the last entry.

**CAM Address Port 2, 1, 0 Registers (CAP2, CAP1, CAP0):** Each CAP is a 16-bit read-only register used to access the CAM cells (Figure 6-13). Each CAM cell is 16 bits wide and contains one third of the 48-bit CAM entry (Figure 6-12) which is used by the SONIC-T for address filtering. The CAP2 register is used to access the upper bits (<47:32>), CAP1 the middle bits (<31:16>) and CAP0 the lower bits (<15:0>) of the CAM entry. Given the physical address 60:50:40:30:20:10, which is made up of 6 octets or bytes, where 10h is the least significant byte and 60h is the most significant byte (60h would be the first byte received from the network and 10h would be the last), CAP0 would be loaded with 2010h, CAP1 with 4030h and CAP2 with 6050h.

To read a CAM entry, the user first places the SONIC-T in software reset (set the RST bit in the Command register), programs the CEP register to select one of sixteen CAM entries, then reads CAP2, CAP1, and CAP0 to obtain the complete 48-bit entry. The user can not write to the CAM entries directly. Instead, the user programs the CAM descriptor area in system memory (see Section 6.1.1), then issues the Load CAM command (setting LCAM bit in the Command register). This causes the SONIC-T to read the descriptors from memory and loads the corresponding CAM entry through CAP2-0.

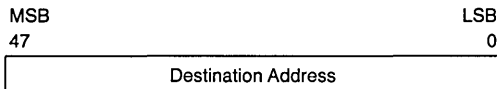


FIGURE 6-12. CAM Entry

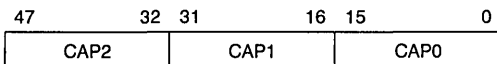


FIGURE 6-13. CAM Address Port Registers

**CAM Enable Register (CE):** The CE is a 16-bit read/write register used to mask out or enable individual CAM entries. Each register bit position corresponds to a CAM entry. When a register bit is set to a "1" the corresponding CAM entry is enabled. When "0" the entry is disabled, this register is unaffected by a software reset and cleared to zero (disabling all entries) during a hardware reset. Under normal operations the user does not access this register. Instead the user sets up this register through the last entry in the CAM descriptor area. The SONIC-T loads the CE register during execution of the LCAM Command.

**CAM Descriptor Pointer Register (CDP):** The CDP is a 15-bit read/write register. The LSB is unused and always

reads back as 0. The CDP is programmed with the lower address (A<15:1>) of the first field of the CAM descriptor block in the CAM descriptor area (CDA) of system memory. SONIC-T uses the contents of the CDP register when accessing the CAM descriptors. This register must be programmed by the user before issuing the LCAM command. During execution of the LCAM Command SONIC-T concatenates the contents of this register with the contents of the URRR register to form the complete 32-bit address. During the Load CAM operation this register is incremented to address the fields in the CDA. After the Load Command completes this register points to the next location after the CAM Descriptor Area.

**CAM Descriptor Count Register (CDC):** The CDC is a 5-bit read/write register. It is programmed with the number of CAM descriptor blocks in the CAM descriptor area. This register must be programmed by the user before issuing the LCAM command. SONIC-T uses the value in this register to determine how many entries to place in the CAM during execution of the LCAM command. During LCAM execution SONIC-T decrements this register each time it reads a descriptor block. When the CDC decrements to zero SONIC-T terminates the LCAM execution. Since the CDC register is programmed with the number of CAM descriptor blocks in the CAM Descriptor Area, the value programmed into the CDC register ranges 1 to 16 (1h to 10h).

### 6.3.11 Tally Counters

The SONIC-T provides three 16-bit counters used for monitoring network statistics on the number of CRC errors, Frame Alignment errors, and missed packets. These registers rollover after the count of FFFFh is reached and produce an interrupt if enabled in the Interrupt Mask Register (IMR). These counters are unaffected by the RXEN bit in the CR, but are halted when the RST bit in the CR is set. The data written to these registers is inverted before being latched. This means that if a value of FFFFh is written to these registers by the system, they will contain and read back the value 0000h. Data is not inverted during a read operation. The Tally registers, therefore, are cleared by writing all "1's" to them. A software or hardware reset does not affect the tally counters.

**CRC Tally Counter Register (CRCT):** The CRCT is a 16-bit read/write register. This register is used to keep track of the number of packets received with CRC errors. After a packet is accepted by the address recognition logic, this register is incremented if a CRC error is detected. If the packet also contains a Frame Alignment error, this counter is not incremented.

**FAE Tally Counter Register (FAET):** The FAET is a 16-bit read/write register. This register is used to keep track of the number of packets received with frame alignment errors. After a packet is accepted by the address recognition logic, this register is incremented if a FAE error is detected.

**Missed Packet Tally Counter Register (MPT):** The MPT is a 16-bit read/write register. After a packet is received, this counter is incremented if there is: (1) lack of memory resources to buffer the packet, (2) a FIFO overrun, or (3) a valid packet has been received, but the receiver is disabled (RXDIS is set in the command register).

## 6.0 SONIC-T Registers (Continued)

### 6.3.12 General Purpose Timer

The SONIC-T contains a 32-bit general-purpose Watchdog Timer for timing user-definable events (*Figure 6-14*). This timer is accessed by the user through two 16-bit read/write registers (WT1 and WT0). The lower count value is programmed through the WT0 register and the upper count value is programmed through the WT1 register.

These two registers are concatenated together to form the complete 32-bit timer. This timer, clocked at  $\frac{1}{2}$  the Transmit Clock (TXC) frequency, counts down from its programmed value and generates an interrupt, if it is enabled Interrupt Mask register, when it rolls over from 0000 0000h to FFFF FFFFh. When the counter rolls over it continues decrementing unless explicitly stopped (setting the STP bit). The timer is controlled by the ST (Start Timer) and STP (Stop Timer) bits in the Command register. A hardware or software reset halts, but does not clear, the General Purpose timer.

31	16	15	0
WT1 (Upper Count Value)		WT0 (Lower Count Value)	

FIGURE 6-14. Watchdog Timer Register

### 6.3.13 Silicon Revision Register

This is a 16-bit read only register. It contains information on the current revision of the SONIC-T. The DP83934BVUL revision register is 0101h.

## 7.0 Bus Interface

SONIC-T features a high speed non-multiplexed address and data bus designed for a wide range of system environments. The data bus can be programmed (via the Data Configuration Register) to a width of either 32- or 16-bits. SONIC-T contains an on-chip DMA and supplies all the necessary signals for DMA operation. With 31 address lines SONIC-T can access a full 2 G-word address space. To accommodate different memory speeds, wait states can be added to the bus cycle by two methods. The memory subsystem can add wait states by simply withholding the appropriate handshake signals or the SONIC-T can be programmed (via the Data Configuration Register) to add wait states.

The SONIC-T is designed to interface to both the National/Intel and Motorola style buses. To facilitate minimum chip

count designs and complete bus compatibility, the user can program the SONIC-T for the following bus modes:

- National/Intel bus operating in synchronous mode
- National/Intel bus operating in asynchronous mode
- Motorola bus operating in synchronous mode
- Motorola bus operating in asynchronous mode

The Bus Mode pin (BMODE) along with the SBUS bit in the Data Configuration Register are used to select the bus mode.

This section illustrates some SONIC-T system interface examples and describes the various SONIC-T bus operations.

### 7.1 PIN CONFIGURATIONS

There are two user selectable pin configurations for SONIC-T to provide the proper interface signals for either the National/Intel or Motorola style buses. The state of the BMODE pin is used to define the pin configuration. Section 1.0 shows the pin configurations for both National/Intel Mode (BMODE = 0, tied to ground) and Motorola Mode (BMODE = 1, tied to  $V_{CC}$ ).

### 7.2 SYSTEM CONFIGURATION

Any device that meets the SONIC-T interface protocol and electrical requirements (timing, threshold, and loading) can be interfaced to SONIC-T. Since two bus protocols are provided, via the BMODE pin, the SONIC-T can interface directly to most microprocessors. *Figure 7-1* shows a typical interface to the National/Intel style bus (BMODE = 0) and *Figure 7-2* shows a typical interface to the Motorola style bus (BMODE = 1).

The BMODE pin also controls byte ordering. When BMODE = 1 big endian byte ordering is selected and when BMODE = 0 little endian byte ordering is selected.

### 7.3 BUS OPERATIONS

There are two types of system bus operations: 1) SONIC-T as a slave, and 2) SONIC-T as a bus master. When SONIC-T is a slave (e.g., a CPU accessing SONIC-T registers) all transfers are non-DMA. When SONIC-T is a bus master (e.g., SONIC-T accessing receive or transmit buffer/descriptor areas) all transfers are block transfers using SONIC-T's on-chip DMA. This section describes the SONIC-T bus operations. Pay special attention to all sections labeled as "Note". These conditions must be met for proper bus operation.

7.0 Bus Interface (Continued)

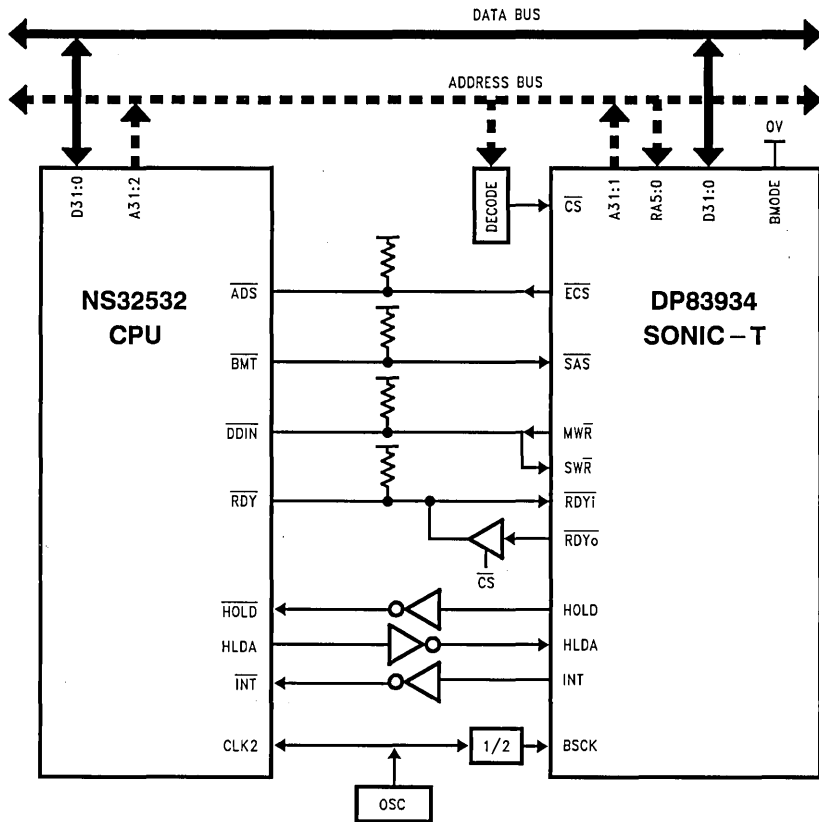


FIGURE 7-1. SONIC-T to NS32532 Interface Example

TL/F/11719-28

7.0 Bus Interface (Continued)

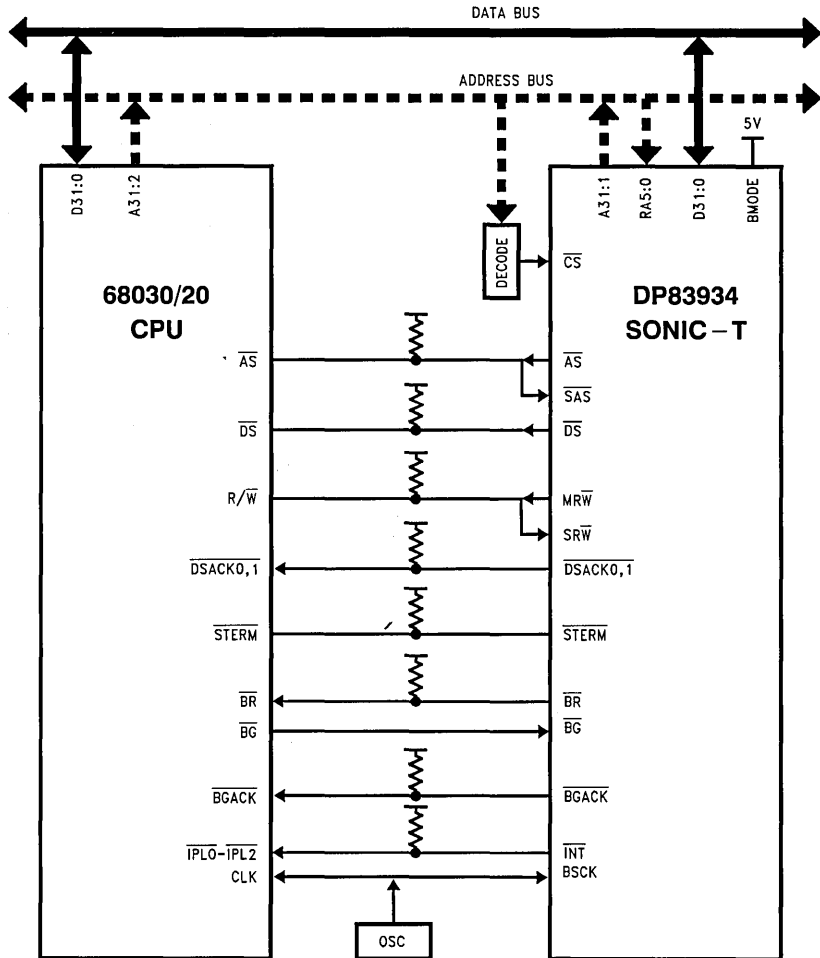


FIGURE 7-2. SONIC-T to Motorola 68030/20 Interface Example

TL/F/11719-29

## 7.0 Bus Interface (Continued)

### 7.3.1 Acquiring The Bus

The SONIC-T requests the bus when 1) its FIFO threshold has been reached or 2) when the descriptor areas in memory (i.e., RRA, RDA, CDA, and TDA) are accessed. Note that when the SONIC-T moves from one area in memory to another (e.g., RBA to RDA), it always deasserts its bus request and then requests the bus again when accessing the next area in memory.

The SONIC-T provides two methods to acquire the bus for compatibility with National/Intel or Motorola type microprocessors. These two methods are selected by setting the proper level on the BMODE pin.

Figures 7-3 and 7-4 show the National/Intel (BMODE = 0) and Motorola (BMODE = 1) bus request timing. Descriptions of each mode follows. For both modes, when the SONIC-T relinquishes the bus, there is an extra holding state (Th) for one bus cycle after the last DMA cycle (T2). This assures that the SONIC-T does not contend with another bus master after it has released the bus.

#### BMODE = 0

The National/Intel processors require a 2-way handshake using a HOLD REQUEST/HOLD ACKNOWLEDGE protocol (Figure 7-3). When the SONIC-T needs to access the bus, it issues a HOLD REQUEST (HOLD) to the microprocessor. The microprocessor, responds with a HOLD ACKNOWLEDGE (HLDA) to the SONIC-T. The SONIC-T then begins its memory transfers on the bus. As long as the CPU maintains HLDA active, the SONIC-T continues until it has finished its memory block transfer. The CPU, however, can preempt the SONIC-T from finishing the block transfer by deasserting HLDA before the SONIC-T deasserts HOLD. This allows a higher priority device to preempt the SONIC-T from continuing to use the bus. The SONIC-T will request the bus again later to complete any operation that it was doing at the time of preemption. The HLDA signal is sampled synchronously by the SONIC-T at the rising edge of the BSCK, setup time must be met to ensure proper operation.

As shown in Figure 7-3, the SONIC-T will assert HOLD to either the falling or rising edge of the bus clock (BSCK). The default is for HOLD to be asserted on the falling edge. Setting the PH bit in the DCR2 (see Section 6.3.7) causes HOLD to be asserted 1/2 bus clock later on the rising edge (shown by the dotted line). Before HOLD is asserted, the SONIC-T checks the HLDA line. If HLDA is asserted, HOLD will not be asserted until after HLDA has been deasserted first.

**Note:** If HLDA is driven low to preempt the SONIC-T from the bus while the SONIC-T is accessing the CAM (LCAM command), the SONIC-T will get off the bus but will not deassert HOLD even though the status bit will indicate idle state. If HLDA is driven low while the SONIC-T is accessing descriptor areas (RRA, RDA, TDA), the SONIC-T will be preempted normally (i.e., get off the bus and deassert HOLD) and the HOLD signal will be reasserted again after one bus clock. If HLDA is driven low while the SONIC-T is accessing data areas (RBA, TBA), the SONIC-T will be preempted normally but may not reassert HOLD unless required to do so depending on the threshold condition of the FIFO.

#### BMODE = 1

The Motorola protocol requires a 3-way handshake using a BUS REQUEST, BUS GRANT, and BUS GRANT ACKNOWLEDGE handshake (Figure 7-4). When using this protocol, the SONIC-T requests the bus by lowering BUS REQUEST  $\overline{BR}$ . The CPU responds by issuing BUS GRANT  $\overline{BG}$ . Upon receiving  $\overline{BG}$ , the SONIC-T assures that all devices have relinquished control of the bus before using the bus. The following signals must be deasserted before the SONIC-T acquires the bus:

- $\overline{BGACK}$
- $\overline{AS}$
- $\overline{DSACK0,1}$
- $\overline{STERM}$  (Asynchronous Mode Only)

Deasserting  $\overline{BGACK}$  indicates that the previous master has released the bus. Deasserting  $\overline{AS}$  indicates that the previous master has completed its cycle and deasserting  $\overline{DSACK0,1}$  and  $\overline{STERM}$  indicates that the previous slave has terminated its connection to the previous master. The SONIC-T maintains its mastership of the bus until it deasserts  $\overline{BGACK}$ . It can not be preempted from the bus.

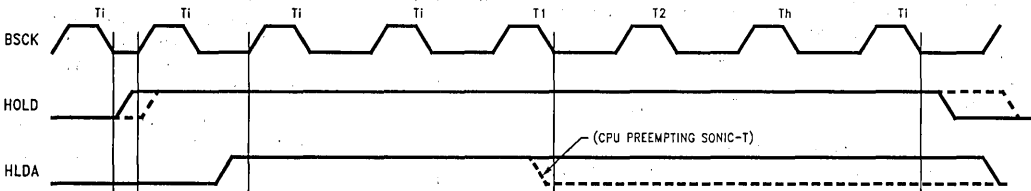


FIGURE 7-3. Bus Request Timing (BMODE = 0)

TL/F/11719-30

## 7.0 Bus Interface (Continued)

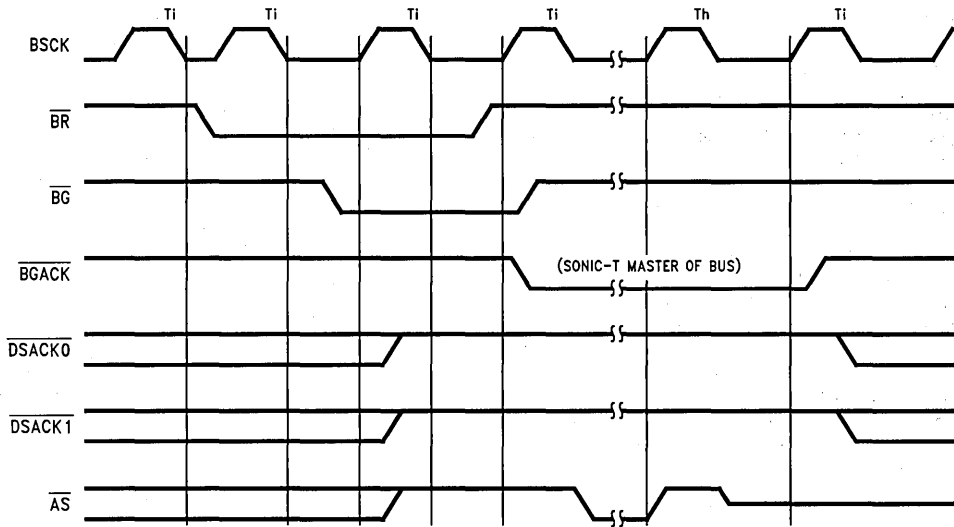


FIGURE 7-4. Bus Request Timing (BMODE = 1)

TL/F/11719-31

### 7.3.2 Block Transfers

The SONIC-T performs block operations during all bus actions, thereby providing efficient transfers to memory. The block cycle consists of three parts. The first part is the bus acquisition phase, as discussed above, in which the SONIC-T gains access to the bus. Once it has access of the bus, the SONIC-T enters the second phase by transferring data to/from its internal FIFOs or registers from/to memory. The SONIC-T transfers data from its FIFOs in either EXACT BLOCK mode or EMPTY/FILL.

**EXACT BLOCK mode:** In this mode the number of words (or long words) transferred during a block transfer is determined by either the Transmit or Receive FIFO thresholds programmed in the Data Configuration Register.

**EMPTY/FILL mode:** In this mode the DMA completely fills the Transmit FIFO during transmission, or completely empties the Receive FIFO during reception. This allows for greater bus latency.

When the SONIC-T accesses the Descriptor Areas (i.e., RRA, RDA, CDA, and TDA), it transfers data between its registers and memory. All fields which need to be used are accessed in one block operation. Thus, the SONIC-T performs 4 accesses in the RRA (see Section 5.4.4.2), 7 accesses in the RDA (see Section 5.4.6.1), 2, 3, or 6 accesses in the TDA (see Section 5.5.4) and 4 accesses in the CDA.

### 7.3.3 Bus Status

The SONIC-T presents three bits of status information on pins S2-S0 which indicate the type of bus operation the SONIC-T is currently performing (Table 7-1). Bus status is valid when at the falling edge of  $\overline{AS}$  or the rising edge of  $\overline{ADS}$ .

TABLE 7-1. Bus Status

S2	S1	S0	Status
1	1	1	The bus is idle. The SONIC-T is not performing any transfers on the bus.
1	0	1	The Transmit Descriptor Area (TDA) is currently being accessed.
0	0	1	The Transmit Buffer Area (TBA) is currently being read.
0	1	1	The Receive Buffer Area (RBA) is currently being written to. Only data is being written, though, not a Source or Destination address.
0	1	0	The Receive Buffer Area (RBA) is currently being written to. Only the Source or Destination address is being written, though.
1	1	0	The Receive Resource Area (RRA) is currently being read.
1	0	0	The Receive Descriptor Area (RDA) is currently being accessed.
0	0	0	The CAM Descriptor Area (CDA) is currently being accessed.

## 7.0 Bus Interface (Continued)

### 7.3.3.1 Bus Status Transitions

When the SONIC-T acquires the bus, it only transfers data to/from a single area in memory (i.e., TDA, TBA, RDA, RBA, RRA, or CDA). Thus, the bus status pins remain stable for the duration of the block transfer cycle with the following three exceptions: 1) if the SONIC-T is accessed during a block transfer, S2-S0 indicates bus idle during the register access, then returns to the previous status, 2) if the SONIC-T finishes writing the Source Address during a block transfer, S2-S0 changes from [0,1,0] to [0,1,1], or 3) during an RDA access between the RXpkt.seq\_no and RXpkt.link access, and between the RXpkt.link and RXpkt.in\_use access, S2-S0 will respectively indicate idle [1,1,1] for 2 or 1 bus clocks. Status will be valid on the falling edge of  $\overline{AS}$  or rising edge of  $\overline{ADS}$ .

Figure 7-5 illustrates the SONIC-T's transitions through memory during the process of transmission and reception. During transmission, the SONIC-T reads the descriptor information from the TDA and then transmits data of the packet from the TBA. The SONIC-T moves back and forth between the TDA and TBA until all fragments and packets are transmitted. During reception, the SONIC-T takes one of two paths. In the first case (path A), when the SONIC-T detects EOL = 0 from the previous reception, it buffers the accepted packet into the RBA, and then writes the descriptor information to the RDA. If the RBA becomes depleted (i.e., RBWC0,1 < EOBC), it moves to the RRA to read a resource descriptor. In the second case (path B), when the SONIC-T detects EOL = 1 from the previous reception, it rereads the RXpkt.link field to determine if the system has

reset the EOL bit since the last reception. If it has, the SONIC-T buffers the packet as in the first case. Otherwise, it rejects the packet and returns to idle.

### 7.3.4 Bus Mode Compatibility

For compatibility with different microprocessor and bus architectures, the SONIC-T operates in one of two modes (set by the BMODE pin) called the National/Intel or little endian mode (BMODE tied low) and the Motorola or big endian mode (BMODE tied high). The definitions for several pins change depending on the mode the SONIC-T is in. Table 7-2 shows these changes. These modes affect both master and slave bus operations with the SONIC-T.

TABLE 7-2. Bus Mode Compatibility

Pin Name	BMODE = 0 (National/Intel)	BMODE = 1 (Motorola)
$\overline{BR}/\text{HOLD}$	HOLD	$\overline{BR}$
$\overline{BG}/\text{HLDA}$	HLDA	$\overline{BG}$
$\text{MR}\overline{W}/\text{M}\overline{W}\overline{R}$	$\text{M}\overline{W}\overline{R}$	$\text{MR}\overline{W}$
$\text{SR}\overline{W}/\text{S}\overline{W}\overline{R}$	$\text{S}\overline{W}\overline{R}$	$\text{SR}\overline{W}$
$\overline{\text{DSACK0}}/\text{RD}\overline{Y}\text{i}$	$\text{RD}\overline{Y}\text{i}$	$\overline{\text{DSACK0}}$
$\overline{\text{DSACK1}}/\text{RD}\overline{Y}\text{o}$	$\text{RD}\overline{Y}\text{o}$	$\overline{\text{DSACK1}}$
$\overline{\text{AS}}/\text{ADS}$	$\overline{\text{ADS}}$	$\overline{\text{AS}}$
$\overline{\text{INT}}/\text{INT}$	INT	$\overline{\text{INT}}$

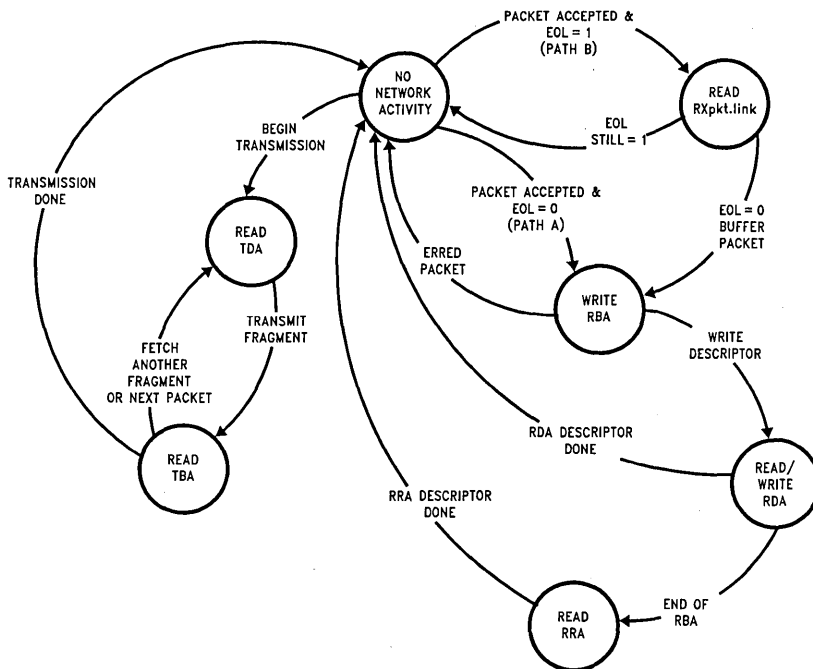


FIGURE 7-5. Bus Status Transitions

TL/F/11719-32

## 7.0 Bus Interface (Continued)

### 7.3.5 Master Mode Bus Cycles

In order to add additional compatibility with different bus architectures, there are two other modes that affect the operation of the bus. These modes are called the synchronous and asynchronous modes and are programmed by setting or resetting the SBUS bit in the Data Configuration Register (DCR). The synchronous and asynchronous modes do not have an effect on slave accesses to the SONIC-T but they do affect the master mode operation. Within the particular bus/processor mode, synchronous and asynchronous modes are very similar. This section discusses all four modes of operation of the SONIC-T (National/Intel vs. Motorola, synchronous vs. asynchronous) when it is a bus master.

In this section, the rising edge of T1 and T2 means the beginning of these states, and the falling edge of T1 and T2 means the middle of these states.

#### 7.3.5.1 Adding Wait States

To accommodate different memory speeds, the SONIC-T provides two methods for adding wait states for its bus operations. Both of these methods can be used singly or in conjunction with each other. A memory cycle is extended by adding additional T2 states. The first method inserts wait-states by withholding the assertion of  $\overline{DSACK0,1}$  or  $\overline{RDYi}$ . The other method allows software to program wait-states. Programming the WC0, WC1 bits in the Data Configuration Register allows 1 to 3 wait-states to be added on each memory cycle. These wait states are inserted between the T1 and T2 bus states and are called T2 (wait) bus states. The SONIC-T will not look at the  $\overline{DSACK0,1}$ ,  $\overline{STERM}$  or  $\overline{RDYi}$  lines until the programmed wait states have passed. Hence, in order to complete a bus operation that

includes programmed wait states, the  $\overline{DSACK0,1}$ ,  $\overline{STERM}$  or  $\overline{RDYi}$  lines must be asserted at their proper times at the end of the cycle during the last T2, not during a programmed wait state. The only exception to this is asynchronous mode where  $\overline{DSACK0,1}$  or  $\overline{RDYi}$  would be asserted during the last programmed wait state, T2 (wait). See the timing for these signals in the timing diagrams for more specific information. Programmed wait states do not affect Slave Mode bus cycles.

#### 7.3.5.2 Memory Cycle for BMODE = 1, Synchronous Mode

On the rising edge of T1, the SONIC-T asserts  $\overline{ECS}$  to indicate that the memory cycle is starting. The address (A31-A1), bus status (S2-S0) and the direction strobe ( $\overline{MRW}$ ) are driven and do not change for the remainder of the memory cycle. On the falling edge of T1, the SONIC-T deasserts  $\overline{ECS}$  and asserts  $\overline{AS}$ .

In synchronous mode,  $\overline{DSACK0,1}$  are sampled on the rising edge of T2. T2 states will be repeated until  $\overline{DSACK0,1}$  are sampled properly in a low state.  $\overline{DSACK0,1}$  must meet the setup and hold times with respect to the rising edge of bus clock for proper operation.

During read cycles (Figure 7-6) data (D31-D0) is latched at the falling edge of T2 and  $\overline{DS}$  is asserted at the falling edge of T1. For write cycles (Figure 7-7) data is driven on the falling edge of T1. If there are wait states inserted,  $\overline{DS}$  is asserted on the falling edge of T2.  $\overline{DS}$  is not asserted for zero wait state write cycles. The SONIC-T terminates the memory cycle by deasserting  $\overline{AS}$  and  $\overline{DS}$  at the falling edge of T2.

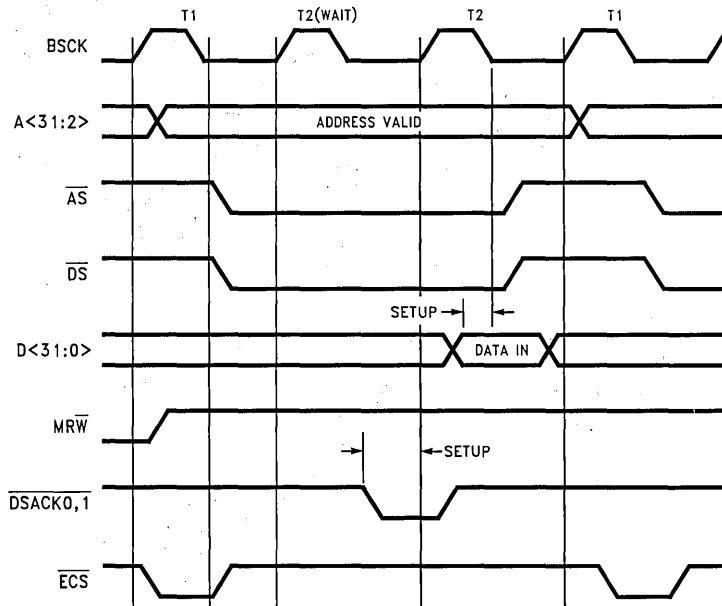
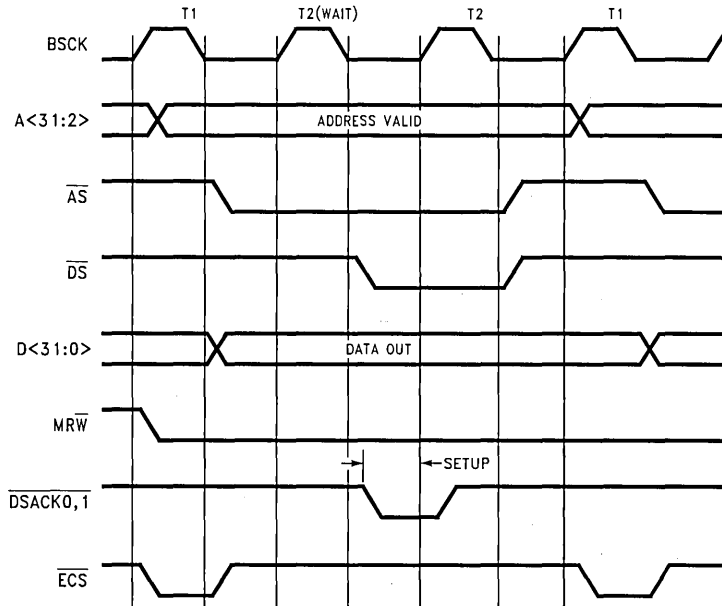


FIGURE 7-6. Memory Read, BMODE = 1, Synchronous (1 Wait-State)

TL/F/11719-33



## 7.0 Bus Interface (Continued)



TL/F/11719-34

FIGURE 7-7. Memory Write, BMODE = 1, Synchronous (1 Wait-State)

### 7.3.5.3 Memory Cycle for BMODE = 1, Asynchronous Mode

On the rising edge of T1, the SONIC-T asserts  $\overline{ECS}$  to indicate that the memory cycle is starting. The address (A31–A1), bus status (S2–S0) and the direction strobe (MRW) are driven and do not change for the remainder of the memory cycle. On the falling edge of T1, the SONIC-T deasserts  $\overline{ECS}$  and asserts  $\overline{AS}$ .

In asynchronous mode,  $\overline{DSACK0,1}$  are asynchronously sampled on the falling edge of both T1 and T2.  $\overline{DSACK0,1}$  do not need to be synchronized to the bus clock because the chip always resolves these signals to either a high or low state. If a synchronous termination of the bus cycle is required, however,  $\overline{STERM}$  may be used.  $\overline{STERM}$  is sampled on the rising edge of T2 and must meet the setup and hold times with respect to that edge for proper operation. Meeting the setup time for  $\overline{DSACK0,1}$  or  $\overline{STERM}$  guarantees that the SONIC-T will terminate the memory cycle 1.5

bus clocks after  $\overline{DSACK0,1}$  were sampled, or 1 cycle after  $\overline{STERM}$  was sampled. T2 states will be repeated until  $\overline{DSACK0,1}$  or  $\overline{STERM}$  are sampled properly in a low state (see note below).

During read cycles (Figures 7-8 and 7-9), data (D31–D0) is latched at the falling edge of T2 and  $\overline{DS}$  is asserted at the falling edge of T1. For write cycles (Figures 7-10 and 7-11) data is driven on the falling edge of T1. If there are wait states inserted,  $\overline{DS}$  is asserted on the falling edge of the first T2 (wait).  $\overline{DS}$  is not asserted for zero wait state write cycles. The SONIC-T terminates the memory cycle by deasserting  $\overline{AS}$  and  $\overline{DS}$  at the falling edge of T2.

**Note:** If the setup time for  $\overline{DSACK0,1}$  is met during T1, or the setup time for  $\overline{STERM}$  is met during the first T2, the full asynchronous bus cycle will take only 2 bus clocks. This may be an unwanted situation. If so,  $\overline{DSACK0,1}$  and  $\overline{STERM}$  should normally be deasserted during T1 and the start of T2 respectively.

## 7.0 Bus Interface (Continued)

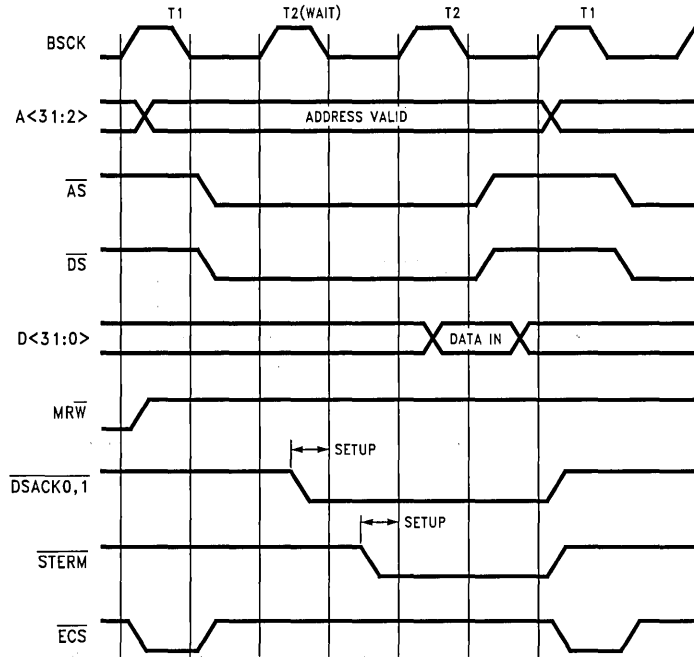


FIGURE 7-8. Memory Read, BMODE = 1, Asynchronous (1 Wait-State)

TL/F/11719-35

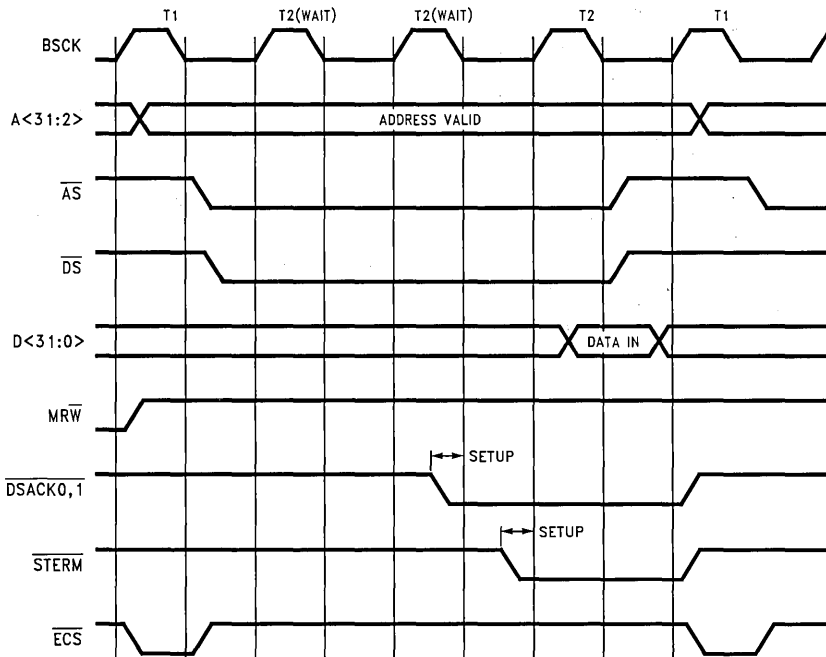
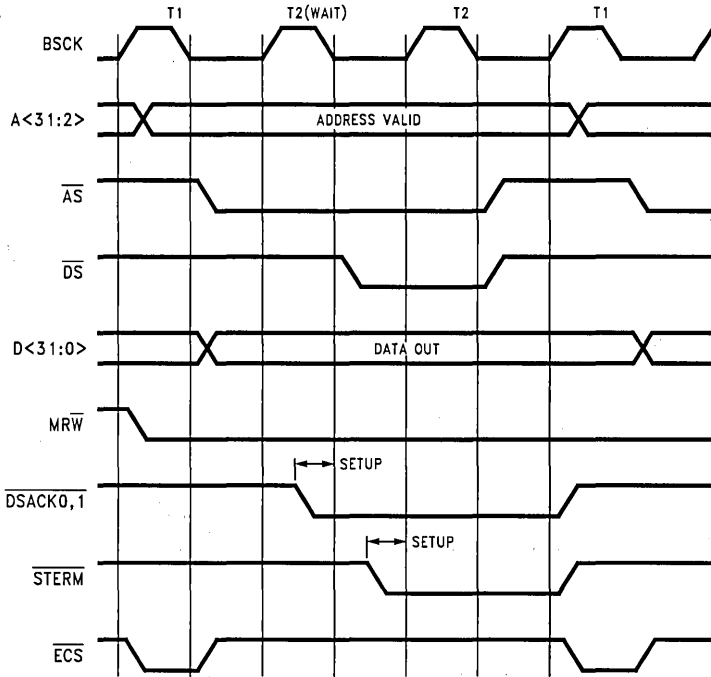


FIGURE 7-9. Memory Read, BMODE = 1, Asynchronous (2 Wait-State)

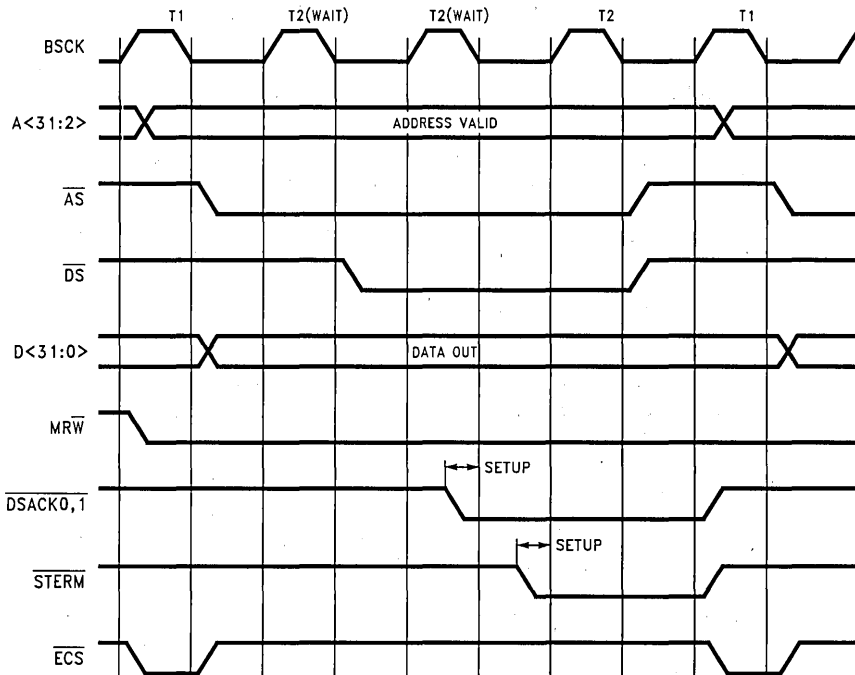
TL/F/11719-36

7.0 Bus Interface (Continued)



TL/F/11719-37

FIGURE 7-10. Memory Write, BMODE = 1, Asynchronous (1 Wait-State)



TL/F/11719-38

FIGURE 7-11. Memory Write, BMODE = 1, Asynchronous (2 Wait-State)

## 7.0 Bus Interface (Continued)

### 7.3.5.4 Memory Cycle for BMODE = 0, Synchronous Mode

On the rising edge of T1, the SONIC-T asserts  $\overline{ADS}$  and  $\overline{ECS}$  to indicate that the memory cycle is starting. The address (A31-A1), bus status (S2-S0) and the direction strobe ( $\overline{MWR}$ ) are driven and do not change for the remainder of the memory cycle. On the falling edge of T1, the SONIC-T deasserts  $\overline{ECS}$ .  $\overline{ADS}$  is deasserted on the rising edge of T2.

In Synchronous mode,  $\overline{RDYi}$  is sampled on the rising edge at the end of T2 (the rising edge of the next T1). T2 states will be repeated until  $\overline{RDYi}$  is sampled properly. In a low state  $\overline{RDYi}$  must meet the setup and hold times with respect to the rising edge of bus clock for proper operation.

During read cycles (Figure 7-12), data (D31-D0) is latched at the rising edge at the end of T2. For write cycles (Figure 7-13), data is driven on the falling edge of T1 and stays driven until the end of the cycle.

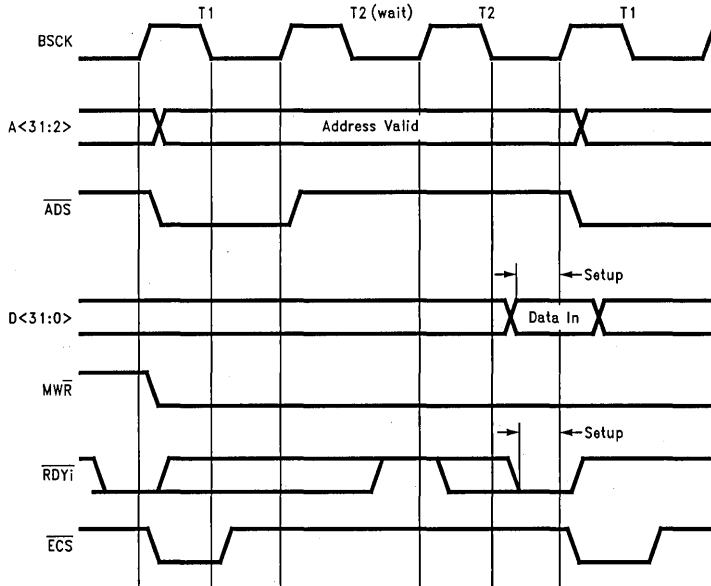


FIGURE 7-12. Memory Read, BMODE = 0, Synchronous (1 Wait-State)

TL/F/11719-39

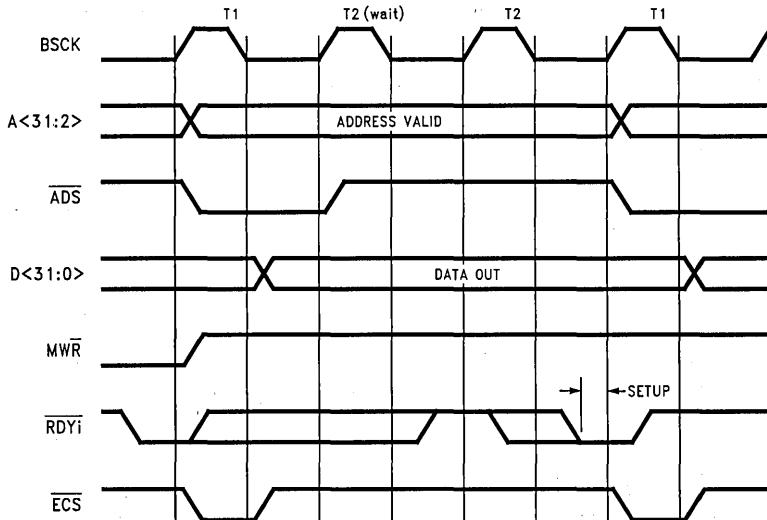


FIGURE 7-13. Memory Write, BMODE = 0, Synchronous (1 Wait-State)

TL/F/11719-40

## 7.0 Bus Interface (Continued)

### 7.3.5.5 Memory Cycle for BMODE = 0, Asynchronous Mode

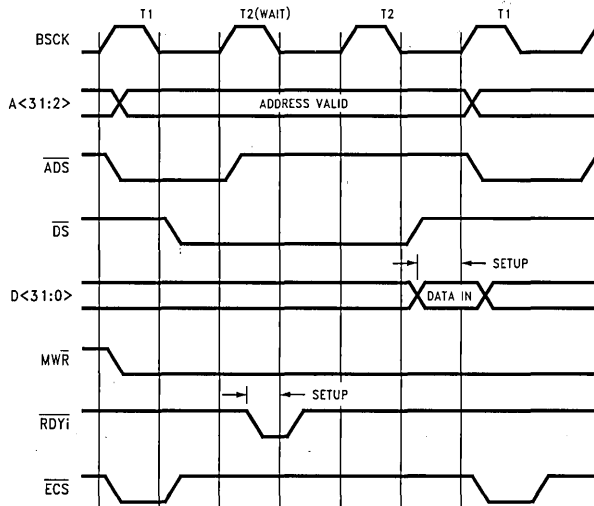
On the rising edge of T1, the SONIC-T asserts  $\overline{ADS}$  and  $\overline{ECS}$  to indicate that the memory cycle is starting. The address (A31-A1), bus status (S2-S0) and the direction strobe ( $\overline{MWR}$ ) are driven and do not change for the remainder of the memory cycle. On the falling edge of T1, the SONIC-T deasserts  $\overline{ECS}$ .  $\overline{ADS}$  is deasserted on the rising edge of T2.

In Asynchronous mode,  $\overline{RDYi}$  is asynchronously sampled on the falling edge of both T1 and T2.  $\overline{RDYi}$  does not need to be synchronized to the bus clock because the chip always resolves these signals to either a high or low state. Meeting the setup time for  $\overline{RDYi}$  guarantees that the

SONIC-T will terminate the memory cycle 1.5 bus clocks after  $\overline{RDYi}$  was sampled. T2 states will be repeated until  $\overline{RDYi}$  is sampled properly in a low state (see note below).

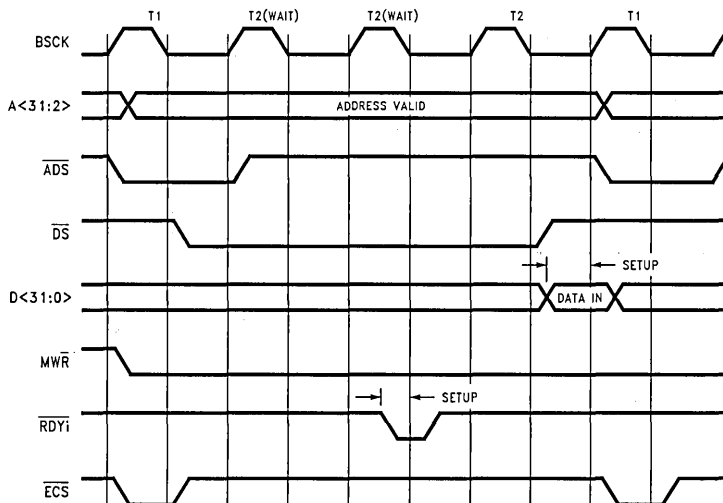
During read cycles (Figures 7-14 and 7-15), data (D31-D0) is latched at the falling edge of T2 and  $\overline{DS}$  is asserted at the rising edge of T1. For write cycles (Figures 7-16 and 7-17) data is driven on the falling edge of T1. If there are wait states inserted,  $\overline{DS}$  is asserted on the rising edge of the first T2 (wait).  $\overline{DS}$  is not asserted for zero wait state write cycles. The SONIC-T terminates the memory cycle by deasserting  $\overline{DS}$  at the falling edge of T2.

**Note:** If the setup time for  $\overline{RDYi}$  is met during T1, the full asynchronous bus cycle will take only 2 bus clocks. This may be an unwanted situation. If so,  $\overline{RDYi}$  should be deasserted during T1.



TL/F/11719-41

FIGURE 7-14. Memory Read, BMODE = 0, Asynchronous (1 Wait-State)



TL/F/11719-42

FIGURE 7-15. Memory Read, BMODE = 0, Asynchronous (2 Wait-State)

7.0 Bus Interface (Continued)

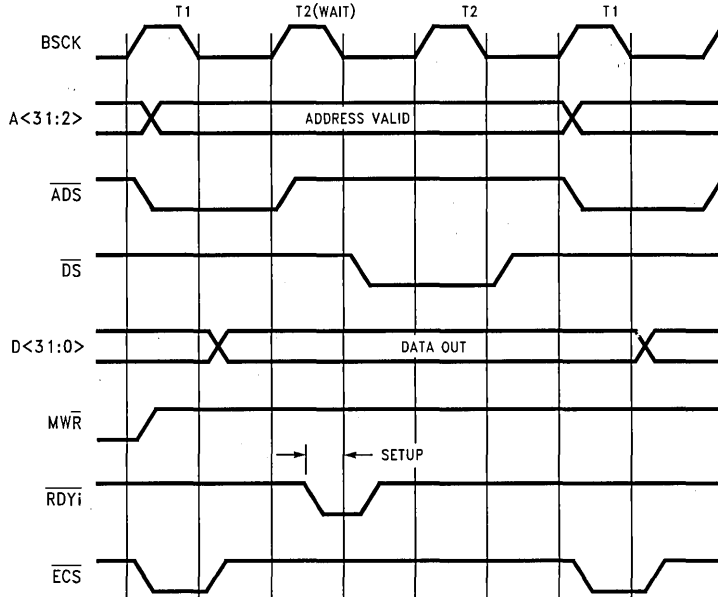


FIGURE 7-16. Memory Write, BMODE = 0, Asynchronous (1 Wait-State)

TL/F/11719-43

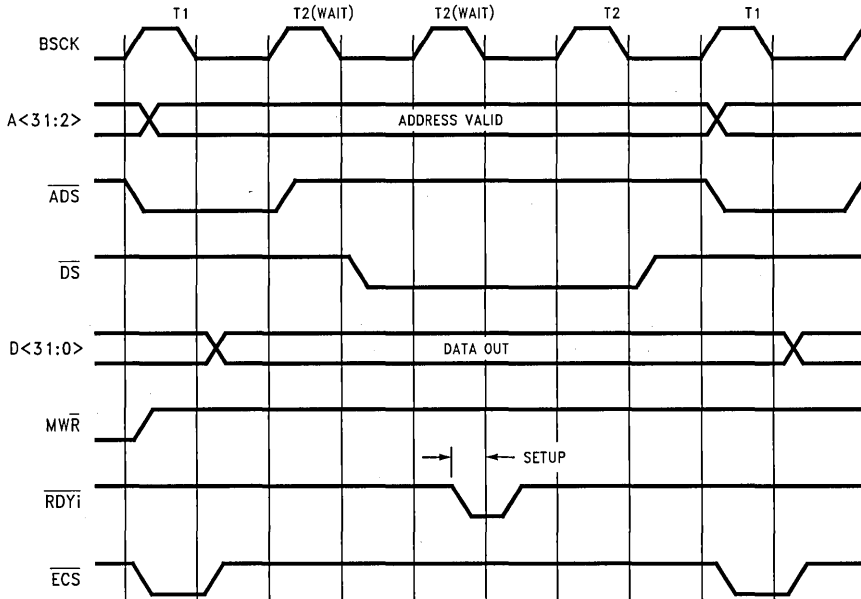


FIGURE 7-17. Memory Write, BMODE = 0, Asynchronous (2 Wait-State)

TL/F/11719-44

## 7.0 Bus Interface (Continued)

### 7.3.6 Bus Exceptions (Bus Retry)

The SONIC-T provides the capability of handling errors during the execution of the bus cycle (Figure 7-18).

The system asserts  $\overline{\text{BRT}}$  (bus retry) to force the SONIC-T to repeat the current memory cycle. When the SONIC-T detects the assertion of  $\overline{\text{BRT}}$ , it completes the memory cycle at the end of T2 and gets off the bus by deasserting  $\overline{\text{BGACK}}$  or HOLD. Then, if Latched Bus Retry mode is not set (LBR in the Data Configuration Register, Section 6.3.2), the SONIC-T requests the bus again to retry the same memory cycle. If Latched Bus Retry is set, though, the SONIC-T will not retry until the BR bit in the ISR (see Section 6.3.6) has been reset and  $\overline{\text{BRT}}$  is deasserted.  $\overline{\text{BRT}}$  has precedence of terminating a memory cycle over  $\overline{\text{DSACK0,1}}$ ,  $\overline{\text{STERM}}$  or  $\overline{\text{RDYi}}$ .

$\overline{\text{BRT}}$  may be sampled synchronously or asynchronously by setting the EXBUS bit in the DCR (see Section 6.3.2). If synchronous Bus Retry is set,  $\overline{\text{BRT}}$  is sampled on the rising edge of T2. If asynchronous Bus Retry is set,  $\overline{\text{BRT}}$  is double synchronized from the falling edge of T1. The asynchronous setup time does not need to be met, but doing so will guarantee that the bus exception will occur in the current bus cycle instead of the next bus cycle. Asynchronous Bus Retry may only be used when the SONIC-T is set to asynchronous mode.

**Note 1:** The deassertion edge of HOLD is dependent on the PH bit in the DCR2 (see Section 6.3.7). Also,  $\overline{\text{BGACK}}$  is driven high for about 0.5 bus clocks before going TRI-STATE.

**Note 2:** If Latched Bus retry is set,  $\overline{\text{BRT}}$  need only satisfy its setup time (the hold time is not important). Otherwise,  $\overline{\text{BRT}}$  must remain asserted until after the Th state.

**Note 3:** If  $\overline{\text{DSACK0,1}}$ ,  $\overline{\text{STERM}}$  or  $\overline{\text{RDYi}}$  remain asserted after  $\overline{\text{BRT}}$ , the next memory cycle, may be adversely affected.

### 7.3.7 Slave Mode Bus Cycle

The SONIC-T's internal registers can be accessed by one of two methods (BMODE = 1 or BMODE = 0). In both methods, the SONIC-T is a slave on the bus. This section describes the SONIC-T's slave mode bus operations.

#### 7.3.7.1 Slave Cycle for BMODE = 1

The system accesses the SONIC-T by driving  $\overline{\text{SAS}}$ ,  $\overline{\text{CS}}$ ,  $\text{SRW}$  and  $\text{RA} < 5:0 >$ . These signals will be sampled each bus cycle, but the SONIC-T will not actually start a slave cycle until  $\overline{\text{CS}}$  has also been asserted.  $\overline{\text{CS}}$  should not be asserted before  $\overline{\text{SAS}}$  is driven low as this will cause improv-

er slave operation. Once  $\overline{\text{SAS}}$  has been driven low, between one and two bus clocks after the assertion of  $\overline{\text{CS}}$ ,  $\overline{\text{SMACK}}$  will be asserted to signify that the SONIC-T has started the slave cycle. Although  $\overline{\text{CS}}$  is an asynchronous input, meeting its setup time (as shown in Figures 7-19 and 7-20) will guarantee that  $\overline{\text{SMACK}}$ , which is asserted off of a falling edge, will be asserted 1 bus clock after the falling edge that  $\overline{\text{CS}}$  is clocked in on. This is assuming that the SONIC-T is not a bus master when  $\overline{\text{CS}}$  was asserted. If the SONIC-T is a bus master, then, when  $\overline{\text{CS}}$  is asserted, the SONIC-T will complete its current master bus cycle and get off the bus temporarily (see Section 7.4.8). In this case,  $\overline{\text{SMACK}}$  will be asserted 5 bus clocks after the falling edge that  $\overline{\text{CS}}$  was clocked in on. This is assuming that there were no wait states in the current master mode access. Wait states will increase the time for  $\overline{\text{SMACK}}$  to go low by the number of wait states in the cycle.

If the slave access is a read cycle (Figure 7-19), then the data will be driven off the same edge as  $\overline{\text{SMACK}}$ . If it is a write cycle (Figure 7-20), then the data will be latched in exactly 2 bus clocks after the assertion of  $\overline{\text{SMACK}}$ . In either case,  $\overline{\text{DSACK0,1}}$  are driven low 2 bus clocks after  $\overline{\text{SMACK}}$  to terminate the slave cycle. For a read cycle, the assertion of  $\overline{\text{DSACK0,1}}$  indicates valid register data and for a write cycle, the assertion indicates that the SONIC-T has latched the data. The SONIC-T deasserts  $\overline{\text{DSACK0,1}}$ ,  $\overline{\text{SMACK}}$  and the data if the cycle is a read cycle at the rising edge of  $\overline{\text{SAS}}$  or  $\overline{\text{CS}}$  depending on which is deasserted first.

**Note 1:** Although the SONIC-T responds as a 32-bit peripheral when it drives  $\overline{\text{DSACK0,1}}$  low, it transfers data only on lines D < 15:0 >.

**Note 2:** For multiple register accesses,  $\overline{\text{CS}}$  can be held low and  $\overline{\text{SAS}}$  can be used to delimit the slave cycle (this is the only case where  $\overline{\text{CS}}$  may be asserted before  $\overline{\text{SAS}}$ ). In this case,  $\overline{\text{SMACK}}$  will be driven low due to  $\overline{\text{SAS}}$  going low since  $\overline{\text{CS}}$  has already been asserted. Notice that this means  $\overline{\text{SMACK}}$  will not stay asserted low during the entire time  $\overline{\text{CS}}$  is low (as is the case for  $\overline{\text{MREQ}}$ , Section 7.3.8).

**Note 3:** If memory request ( $\overline{\text{MREQ}}$ ) follows a chip select ( $\overline{\text{CS}}$ ), it must be asserted at least 2 bus clocks after  $\overline{\text{CS}}$  is deasserted. Both  $\overline{\text{CS}}$  and  $\overline{\text{MREQ}}$  must not be asserted concurrently.

**Note 4:** When  $\overline{\text{CS}}$  is deasserted, it must remain deasserted for at least one bus clock.

**Note 5:** The way in which  $\overline{\text{SMACK}}$  is asserted due to  $\overline{\text{CS}}$  is not the same as the way in which  $\overline{\text{SMACK}}$  is asserted due to  $\overline{\text{MREQ}}$ . The assertion of  $\overline{\text{SMACK}}$  is dependent upon both  $\overline{\text{CS}}$  and  $\overline{\text{SAS}}$  being low, not just  $\overline{\text{CS}}$ . This is not the same as the case for  $\overline{\text{MREQ}}$  (see Section 7.3.8). The assertion of  $\overline{\text{SMACK}}$  in these two cases should not be confused.

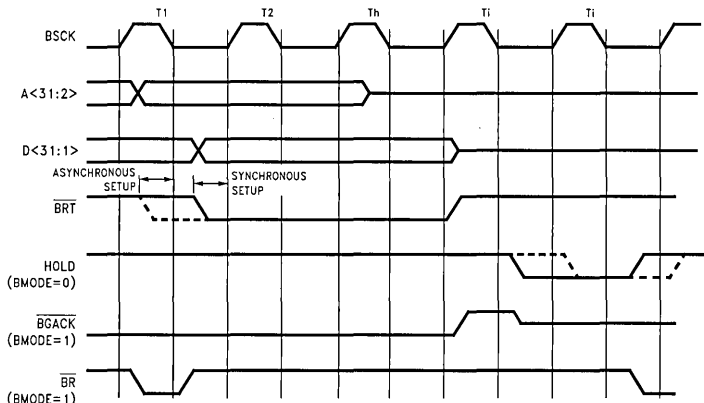


FIGURE 7-18. Bus Exception (Bus Retry)

TL/F/11719-45

### 7.0 Bus Interface (Continued)

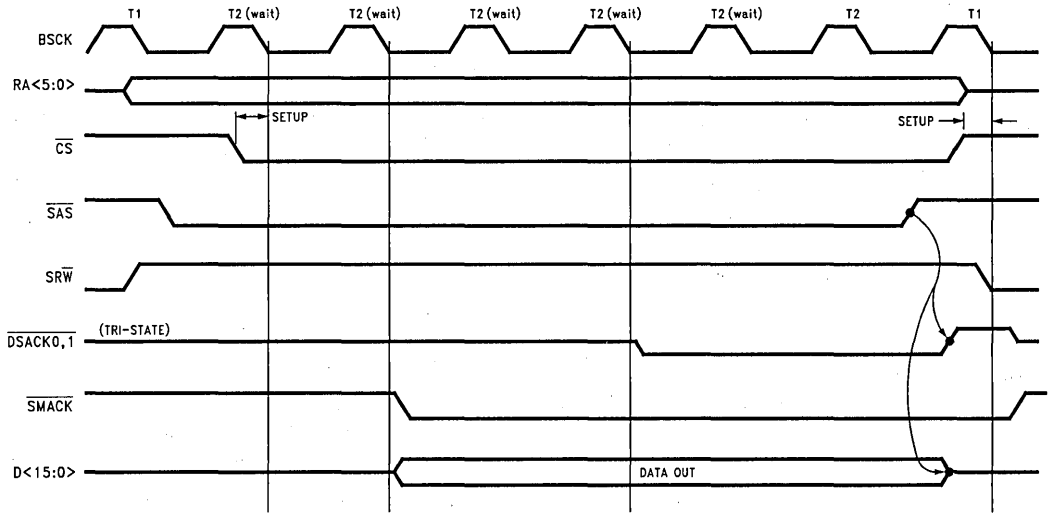


FIGURE 7-19. Register Read, BMODE = 1

TL/F/11719-46

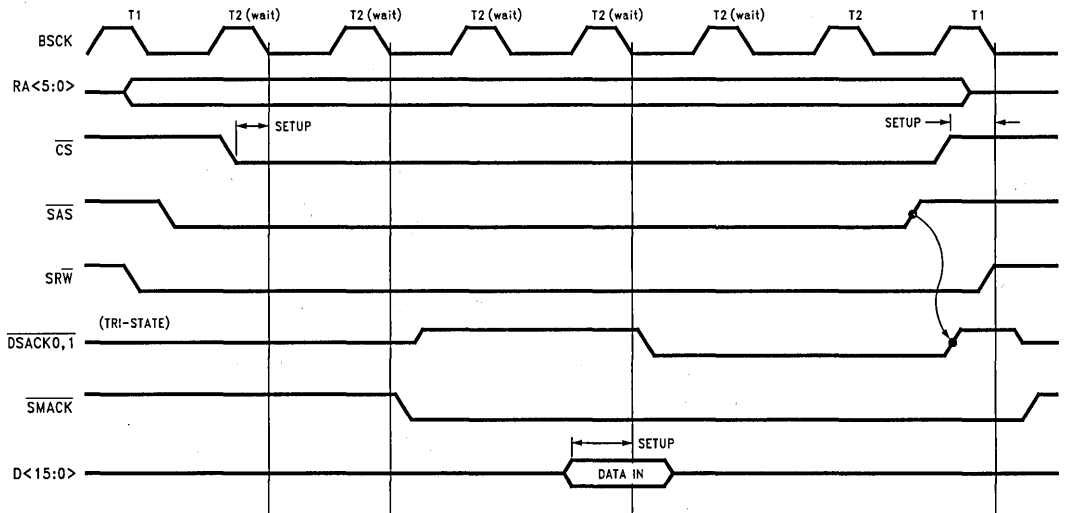


FIGURE 7-20. Register Write, BMODE = 1

TL/F/11719-47



## 7.0 Bus Interface (Continued)

### 7.3.7.2 Slave Cycle for BMODE = 0

The system accesses the SONIC-T by driving  $\overline{SAS}$ ,  $\overline{CS}$ ,  $\overline{SWR}$  and  $RA<5:0>$ . These signals will be sampled each bus cycle, but the SONIC-T will not actually start a slave cycle until  $\overline{CS}$  has been sampled low and  $\overline{SAS}$  has been sampled high.  $\overline{CS}$  should not be asserted low before the falling edge of  $\overline{SAS}$  as this will cause improper slave operation.  $\overline{CS}$  may be asserted low, however, before the rising edge of  $\overline{SAS}$ . In this case, it is suggested that  $\overline{SAS}$  be driven high within one bus clock after the falling edge of  $\overline{CS}$ . Between one and two bus clocks after the assertion of  $\overline{CS}$ , once  $\overline{SAS}$  has been driven high,  $\overline{SMACK}$  will be driven low to signify that the SONIC-T has started the slave cycle. Although  $\overline{CS}$  is an asynchronous input, meeting its setup time (as shown in Figures 7-21 and 7-22) will guarantee that  $\overline{SMACK}$ , which is asserted off a falling edge, will be asserted 1 bus clock after the falling edge that  $\overline{CS}$  was clocked in on. This is assuming that the SONIC-T is not a bus master when  $\overline{CS}$  is asserted. If the SONIC-T is a bus master, then, when  $\overline{CS}$  is asserted, the SONIC-T will complete its current master bus cycle and get off the bus temporarily (see Section 7.3.8). In this case,  $\overline{SMACK}$  will be asserted 5 bus clocks after the falling edge that  $\overline{CS}$  was clocked in on. This is assuming that there were no wait states in the current master mode access. Wait states will increase the time for  $\overline{SMACK}$  to go low by the number of wait states in the cycle.

If the slave access is a read cycle (Figure 7-21), then the data will be driven off the same edge as  $\overline{SMACK}$ . If it is a write cycle (Figure 7-22), then the data will be latched in exactly 2 bus clocks after the assertion of  $\overline{SMACK}$ . In either case,  $\overline{RDY0}$  is driven low 2.5 bus clocks after  $\overline{SMACK}$  to terminate the slave cycle. For a read cycle, the assertion of  $\overline{RDY0}$  indicates valid register data and for a write cycle, the assertion indicates that the SONIC-T has latched the data. The SONIC-T deasserts  $\overline{RDY0}$ ,  $\overline{SMACK}$  and the data if the cycle is a read cycle at the falling edge of  $\overline{SAS}$  or the rising edge of  $\overline{CS}$  depending on which is first.

**Note 1:** The SONIC-T transfers data only on lines  $D<15:0>$  during slave mode accesses.

**Note 2:** For multiple register accesses,  $\overline{CS}$  can be held low and  $\overline{SAS}$  can be used to delimit the slave cycle (this is the only case where  $\overline{CS}$  may be asserted before  $\overline{SAS}$ ). In this case,  $\overline{SMACK}$  will be driven low due to  $\overline{SAS}$  going high since  $\overline{CS}$  has already been asserted. Notice that this means  $\overline{SMACK}$  will not stay asserted low during the entire time  $\overline{CS}$  is low (as is the case for  $\overline{MREQ}$ , Section 7.3.8).

**Note 3:** If memory request ( $\overline{MREQ}$ ) follows a chip select  $\overline{CS}$ , it must be asserted at least 2 bus clocks after  $\overline{CS}$  is deasserted. Both  $\overline{CS}$  and  $\overline{MREQ}$  must not be asserted concurrently.

**Note 4:** When  $\overline{CS}$  is deasserted, it must remain deasserted for at least one bus clock.

**Note 5:** The way in which  $\overline{SMACK}$  is asserted due to  $\overline{CS}$  is not the same as the way in which  $\overline{SMACK}$  is asserted due to  $\overline{MREQ}$ . The assertion of  $\overline{SMACK}$  is dependent upon both  $\overline{CS}$  and  $\overline{SAS}$  being low, not just  $\overline{CS}$ . This is not the same as the case for  $\overline{MREQ}$  (see Section 7.3.8). The assertion of  $\overline{SMACK}$  in these two cases should not be confused.

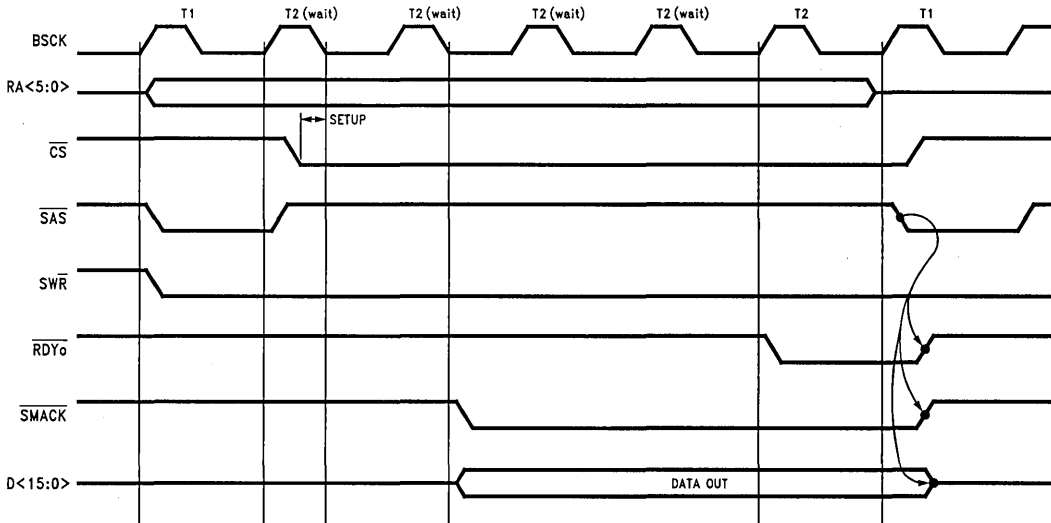


FIGURE 7-21. Register Read, BMODE = 0

TL/F/11719-48

### 7.0 Bus Interface (Continued)

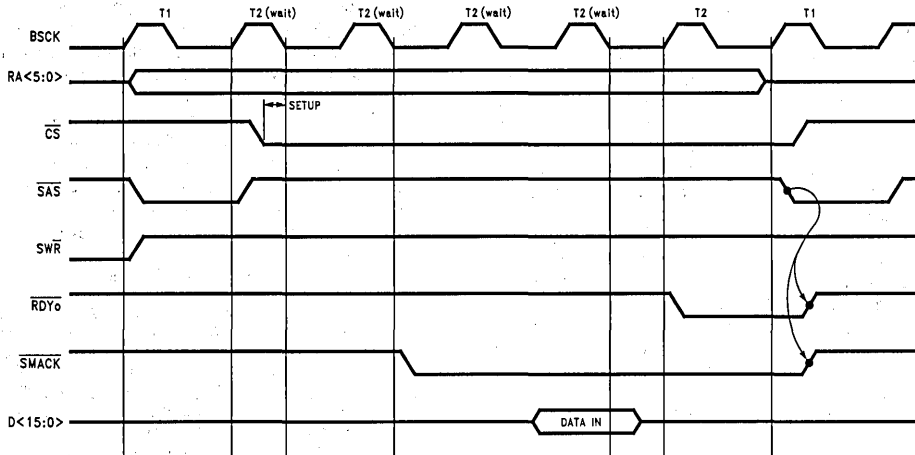


FIGURE 7-22. Register Write, BMODE = 0

TL/F/11719-49

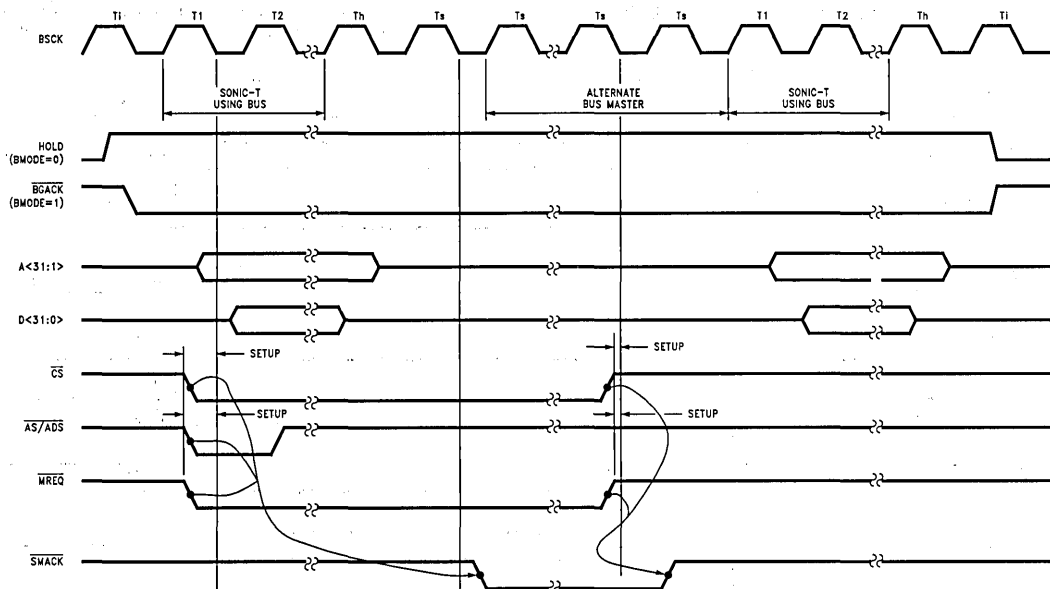


FIGURE 7-23. On-Chip Memory Arbitrer

TL/F/11719-50

## 7.0 Bus Interface (Continued)

### 7.3.8 On-Chip Memory Arbiter

For applications which share the buffer memory area with the host system (shared-memory applications), the SONIC-T provides a fast on-chip memory arbiter for efficiently resolving accesses between the SONIC-T and the host system (Figure 7-23). The host system indicates its intentions to use the shared-memory by asserting Memory Request (MREQ). The SONIC-T will allow the host system to use the shared memory by acknowledging the host system's request with Slave and Memory Acknowledge SMACK. Once SMACK is asserted, the host system may use the shared memory freely. The host system gives up the shared memory by deasserting MREQ.

MREQ is clocked in on the falling edge of bus clock and is double synchronized internally to the rising edge. SMACK is asserted on the falling edge of a Ts bus cycle. If the SONIC-T is not currently accessing the memory, SMACK is asserted immediately after MREQ was clocked in. If, however, the SONIC-T is accessing the shared memory, it finishes its current memory transfer and then issues SMACK. SMACK will be asserted 1 or 5 bus clocks, respectively, after MREQ is clocked in. Since MREQ is double synchronized, it is not necessary to meet its setup time. Meeting the setup time for MREQ will, however, guarantee that SMACK is asserted in the next or fifth bus clock after the current bus clock. SMACK will deassert within one bus clock after MREQ is deasserted. The SONIC-T will then finish its master operation if it was using the bus previously.

If the host system needs to access the SONIC-T's registers instead of shared memory, CS would be asserted instead of MREQ. Accessing the SONIC-T's registers works almost exactly the same as accessing the shared memory except that the SONIC-T goes into a slave cycle instead of going idle. See Section 7.3.7 for more information about how register accesses work.

**Note 1:** The successive assertion of CS and MREQ must be separated by at least two bus clocks. Both CS and MREQ must not be asserted concurrently.

**Note 2:** The number of bus clocks between MREQ being asserted and the assertion of SMACK when the SONIC-T is in Master Mode is 5 bus clocks assuming there were no wait states in the Master Mode access. Wait states will increase the time for SMACK to go low by the number of wait states in the cycle (the time will be 5 + the number of wait states).

**Note 3:** The way in which SMACK is asserted is due to CS is not the same as the way in which SMACK is asserted due to MREQ. SMACK goes low as a direct result of the assertion of MREQ, whereas, for CS, SAS must also be driven low (BMODE = 1) or high (BMODE = 0) before SMACK will be asserted. This means that when SMACK is asserted due to MREQ, SMACK will remain asserted until MREQ is deasserted. Multiple memory accesses can be made to the shared memory without SMACK ever going high. When SMACK is asserted due to CS, however, SMACK will only remain low as long as SAS is also low (BMODE = 1) or high (BMODE = 0). SMACK will not remain low throughout multiple register accesses to the SONIC-T because SAS must toggle for each register access. This is an important difference to consider when designing shared memory designs.

### 7.3.9 Chip Reset

The SONIC-T has two reset modes; a hardware reset and a software reset. The SONIC-T can be hardware reset by asserting the RESET pin or software reset by setting the RST bit in the Command Register (Section 6.3.1). The two reset modes are not interchangeable since each mode performs a different function.

After power-on, the SONIC-T must be hardware reset before it will become operational. This is done by asserting RESET for a minimum of 10 transmit clocks (10 ethernet transmit clock periods, TXC). If the bus clock (BSCK) period is greater than the transmit clock period, RESET should be asserted for 10 bus clocks instead of 10 transmit clocks. A hardware reset places the SONIC-T in the following state. (The registers affected are listed in parentheses. See Table 7-3 and Section 6.3 for more specific information about the registers and how they are affected by a hardware reset. Only those registers listed below and in Table 7-3 are affected by a hardware reset.)

1. Receiver and Transmitter are disabled (CR).
2. The General Purpose timer is halted (CR).
3. All interrupts are masked out (IMR).
4. The NCRS and PTX status bits in the Transmit Control Register (TCR) are set.
5. The End Of Byte Count (EOBC) register is set to 02F8h (760 words).
6. Packet and buffer sequence number counters are set to zero.
7. All CAM entries are disabled. The broadcast address is also disabled (CAM Enable Register and the RCR).
8. Loopback operation is disabled (RCR).
9. The latched bus retry is set to the unlatched mode (DCR).
10. All interrupt status bits are reset (ISR).
11. The Extended Bus Mode is disabled (DCR).
12. HOLD will be asserted/deasserted from the falling clock edge (DCR2).

**TABLE 7-3. Internal Register Content after Reset**

Register	Contents after Reset	
	Hardware Reset	Software Reset
Command	0094h	0094h/00A4h
Data Configuration (DCR and DCR2)	*	unchanged
Interrupt Mask	0000h	unchanged
Interrupt Status	0000h	unchanged
Transmit Control	0101h	unchanged
Receive Control	**	unchanged
End Of Buffer Count	02F8h	unchanged
Sequence Counters	0000h	unchanged
CAM Enable	0000h	unchanged

\*Bits 15 and 13 of the DCR and bits 4 through 0 of the DCR2 are reset to a 0 during a hardware reset. Bits 15–12 of the DCR2 are unknown until written to. All other bits in these two registers are unchanged.

\*\*Bits LB1, LB0 and BRD are reset to a 0 during hardware reset. All other bits are unchanged.

### 7.0 Bus Interface (Continued)

- 13. PCOMP will not be asserted (DCR2).
- 14. Packets will be accepted (not rejected) on CAM match (DCR2).

A software reset immediately terminates DMA operations and future interrupts. The chip is put into an idle state where registers can be accessed, but the SONIC-T will not be active in any other way. The registers are affected by a software reset as shown in Table 7-3 (only the Command Register is changed).

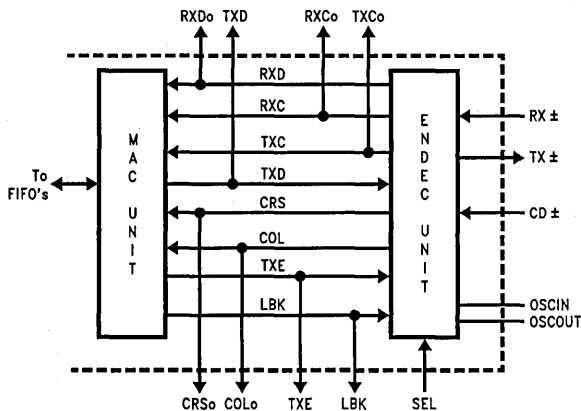
### 8.0 Network Interfacing

The SONIC-T contains an on-chip ENDEC that performs the network interfacing between the AUI (Attachment Unit Inter-

face) and the SONIC-T's MAC unit. A pin selectable option allows the internal ENDEC to be disabled and the MAC/ENDEC signals to be supplied to the user for connection to an external ENDEC. If the EXT pin is tied to ground (EXT=0) the internal ENDEC is selected and if EXT is tied to V<sub>CC</sub> (EXT=1) the external ENDEC option is selected.

**Internal ENDEC:** When the internal ENDEC is used (EXT=0) the interface signals between the ENDEC and MAC unit are internally connected. While these signals are used internally by the SONIC-T they are also provided as an output to the user (Figure 8-1).

The internal ENDEC allows for a 2-chip solution for the complete Ethernet interface. Figure 8-2 shows a typical diagram of the Thin Ethernet and AUI network interface.



TL/F/11719-51

FIGURE 8-1. MAC and Internal ENDEC Interface Signals

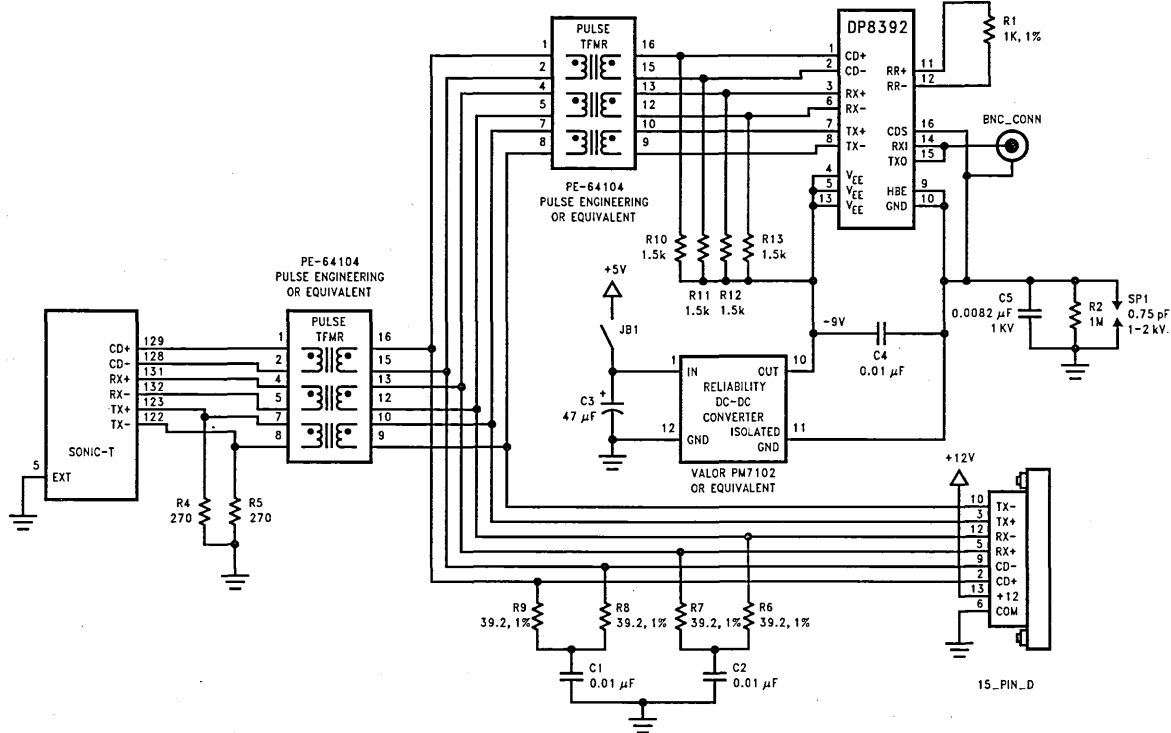


FIGURE 8-2. Network Interface Example (EXT = 0)

TL/F/11719-52

## 8.0 Network Interfacing (Continued)

**External ENDEC:** When EXT = 1 the internal ENDEC is bypassed and the signals are provided directly to the user. Since SONIC-T's on-chip ENDEC is the same as National's DP83910 Serial Network Interface (SNI) the interface considerations discussed in this section would also apply to using this device in the external ENDEC mode.

### 8.1 MANCHESTER ENCODER AND DIFFERENTIAL DRIVER

The ENDEC unit's encoder begins operation when the MAC section begins sending the serial data stream. It converts NRZ data from the MAC section to Manchester data for the differential drivers (TX±). In Manchester encoding, the first half of the bit cell contains the complementary data and the second half contains the true data (Figure 8-3). A transition always occurs at the middle of the bit cell. As long as the MAC continues sending data, the ENDEC section remains in operation. At the end of transmission, the last transition is always positive, occurring at the center of the bit cell if the last bit is a one, or at the end of the bit cell if the last bit is a zero.

The differential transmit pair drives up to 50 meters of twisted pair AUI cable. These outputs are source followers which require two 270Ω pull-down resistors to ground. In addition, a pulse transformer is required between the transmit pair output and the AUI interface.

The driver provides full-step mode for compatibility with Ethernet and IEEE 802.3, so that TX+ and TX- are equal in the idle state.

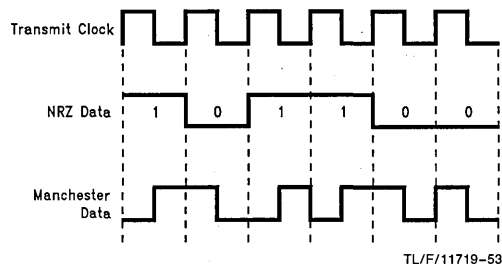


FIGURE 8.3. Manchester Encoded Data Stream

#### 8.1.1 Manchester Decoder

The decoder consists of a differential receiver and a phase lock loop (PLL) to separate the Manchester encoded data stream into clock signals and NRZ data. The differential input must be externally terminated with two 39Ω resistors connected in series. In addition, a pulse transformer is required between the receive input pair and the AUI interface.

To prevent noise from falsely triggering the decoder, a squelch circuit at the input rejects signals with a magnitude less than -175 mV. Signals more negative than -300 mV are decoded.

Once the input exceeds the squelch requirements, the decoder begins operation. The decoder may tolerate bit jitter up to 18 ns in the received data. The decoder detects the end of a frame within one and a half bit times after the last bit of data.

#### 8.1.2 Collision Translator

When the Ethernet transceiver (DP8392 CTI) detects a collision, it generates a 10 MHz signal to the differential collision inputs (CD+ and CD-) of the SONIC-T. When SONIC-T

detects these inputs active, its Collision translator converts the 10 MHz signal to an active collision signal to the MAC section. This signal causes SONIC-T to abort its current transmission and reschedule another transmission attempt.

The collision differential inputs are terminated the same way as the differential receive inputs and a pulse transformer is required between the collision input pair and the AUI interface. The squelch circuitry is also similar, rejecting pulses with magnitudes less than -175 mV.

#### 8.1.3 Oscillator Inputs

The oscillator inputs to the SONIC-T (OSCIN and OSCOUT) can be driven with a parallel resonant crystal or an external clock. In either case the oscillator inputs must be driven with a 20 MHz signal. The signal is divided by 2 to generate the 10 MHz transmit clock (TXC) for the MAC unit. The oscillator also provides internal clock signals for the encoding and decoding circuits.

##### 8.1.3.1 External Crystal

According to the IEEE 802.3 standard, the transmit clock (TXC) must be accurate to 0.01%. This means that the oscillator circuit, which includes the crystal and other parts involved must be accurate to 0.01% after the clock has been divided in half. Hence, when using a crystal, it is necessary to consider all aspects of the crystal circuit. An example of a recommended crystal circuit is shown in Figure 8-4 and suggested oscillator specifications are shown in Table 8-1. The load capacitors in Figure 8-4, C1 and C2, should be no greater than 36 pF each, including all stray capacitance (see note 2). The resistor, R1, may be required in order to minimize frequency drift due to changes in V<sub>CC</sub>. If R1 is required, its value must be carefully selected since R1 decreases the loop gain. If R1 is made too large, the loop gain will be greatly reduced and the crystal will not oscillate. If R1 is made too small, normal variations in V<sub>CC</sub> may cause the oscillation frequency to drift out of specification. As a first rule of thumb, the value of R1 should be made equal to five times the motional resistance of the crystal. The motional resistance of 20 MHz crystals is usually in the range of 10Ω to 30Ω. This implies that reasonable values for R1 should be in the range of 50Ω to 150Ω. The decision of whether or not to include R1 should be based upon measured variations of crystal frequency as each of the circuit parameters are varied.

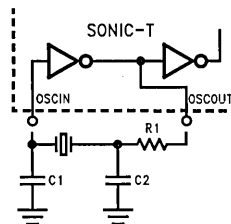


FIGURE 8-4. Crystal Connection to the SONIC-T  
(see text)

**Note 1:** The OSCOUT pin is not guaranteed to provide a TTL compatible logic output, and should not be used to drive any external logic. If additional logic needs to be driven, then an external oscillator should be used as described in the following section.

**Note 2:** The frequency marked on the crystal is usually measured with a fixed load capacitance specified in the crystal's data sheet. The actual load capacitance used should be the specified value minus the stray capacitance.

## 8.0 Network Interfacing (Continued)

**TABLE 8-1. Crystal Specifications**

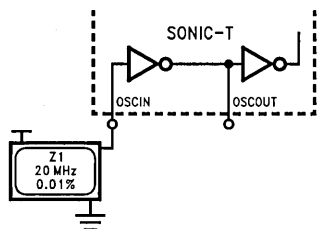
Resonant frequency	20 MHz
Tolerance (see text)	±0.01% at 25°C
Accuracy	±0.005% (50 ppm) at 0 to 70°C
Fundamental Mode Series Resistance	≤25Ω
Specified Load Capacitance	≤18 pF
Type	AT cut
Circuit	Parallel Resonance

### 8.1.3.2 Clock Oscillator Module

The SONIC-T also allows an external clock oscillator to be used. The connection configuration is shown in *Figure 8-5*. This connection requires an oscillator with the following specifications:

1. TTL or CMOS output with a 0.01% frequency tolerance
2. 40%–60% duty cycle
3. One CMOS load output drive

These specifications assume that no other circuitry is driven. In this configuration the OSCOUT pin must be left open.



TL/F/11719-55

**FIGURE 8-5. Oscillator Module Connection to the SONIC-T**

### 8.1.3.3 PCB Layout Considerations

Care should be taken when connecting a crystal. Stray capacitance (e.g., from PC board traces and plated through-holes around the OSCIN and OSCOUT pins) can shift the crystal's frequency out of range, causing the transmitted frequency to exceed the 0.01% tolerance specified by IEEE. The layout considerations for using an external crystal are rather straightforward. The oscillator layout should locate all components close to the OSCIN and OSCOUT pins and

should use short traces that avoid excess capacitance and inductance. A solid ground should be used to connect the ground legs of the two capacitors.

When connecting an external oscillator, the only considerations are to keep the oscillator module as close to the SONIC-T as possible to reduce stray capacitance and inductance and to give the module a clean  $V_{CC}$  and a solid ground.

### 8.1.4 Power Supply Considerations

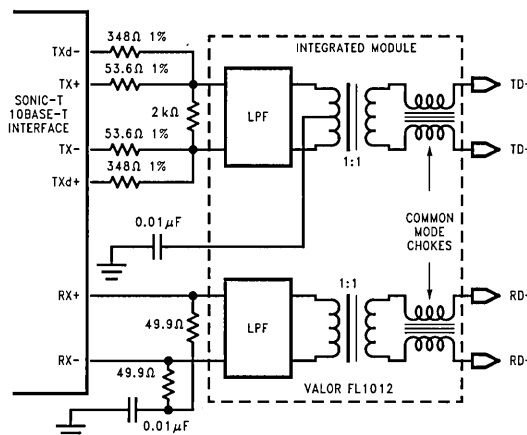
In general, power supply routing and design for the SONIC-T need only follow standard practices. In some situations, however, additional care may be necessary in the layout of the analog supply. Specifically special care may be needed for the  $TXV_{CC}$ ,  $RXV_{CC}$ ,  $PLL_{VCC}$ ,  $OSCV_{CC}$ ,  $RXTV_{CC}$  and  $TPV_{CC}$  power supplies and the  $TXGND$ ,  $RXGND$ ,  $PLL_{GND}$ ,  $OSCGND$ ,  $TPGND$  and  $ANGND$ . In most cases the analog and digital power supplies can be interconnected. However, to ensure optimum performance of the SONIC-T's analog functions, power supply noise should be minimized. To reduce analog supply noise, any of several techniques can be used.

1. Route analog supplies as a separate set of traces or planes from the digital supplies with their own decoupling capacitors.
2. Provide noise filtering on the analog supply pins by inserting a low pass filter. Alternatively, a ferrite bead could be used to reduce high frequency power supply noise.
3. Utilize a separate regulator to generate the analog supply.

## 8.2 TWISTED PAIR INTERFACE MODULE

**Transmitter Considerations:** The transmitter consists of four signals, the true and complement Manchester encoded data ( $TXO_{\pm}$ ) and these signals delayed by 50 ns ( $TXOd_{\pm}$ ).

These four signals are resistively combined (*Figure 8-6*),  $TXO_{+}$  with  $TXOd_{-}$  and  $TXO_{-}$  with  $TXOd_{+}$ , in a configuration referred to as pre-emphasis. This digital pre-emphasis is required to compensate for the low-pass filter effects of the twisted pair cable, which cause greater attenuation to the 10 MHz (50 ns) pulses of the Manchester encoded waveform than the 5 MHz (100 ns) pulses.



TL/F/11719-56

**FIGURE 8-6. External Circuitry to Connect the SONIC-T to Twisted Pair Cable**

## 9.0 AC and DC Specifications

### Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage ( $V_{CC}$ )	-0.5V to 7.0V
DC Input Voltage ( $V_{IN}$ )	-0.5V to $V_{CC} + 0.5V$
DC Output Voltage ( $V_{OUT}$ )	-0.5V to $V_{CC} + 0.5V$

Storage Temperature Range ( $T_{STG}$ )	-65°C to 150°C
Power Dissipation (PD)	500 mW
Lead Temp. (TL) (Soldering, 10 sec.)	260°C
ESD Rating ( $R_{ZAP} = 1.5k, C_{ZAP} = 120 pF$ )	1.5 kV

### DC Specifications $T_A = 0^\circ\text{C}$ to $70^\circ\text{C}$ , $V_{CC} = 5V \pm 5\%$ unless otherwise specified

Symbol	Parameter	Conditions	Min	Max	Units
$V_{OH}$	Minimum High Level Output Voltage	$I_{OH} = -8 \text{ mA}$	3.0		V
$V_{OL}$	Maximum Low Level Output Voltage	$I_{OL} = 8 \text{ mA}$		0.5	V
$V_{IH}$	Minimum High Level Input Voltage		2.0		V
$V_{IL}$	Maximum Low Level Input Voltage			0.8	V
$I_{IN}$	Input Current	$V_{IN} = V_{CC}$ or GND	-10	10	$\mu\text{A}$
$I_{OZ}$	TRI-STATE Output Leakage Current	$V_{OUT} = V_{CC}$ or GND	-10	10	$\mu\text{A}$
$I_{CC}$	Average Operating Supply Current	$I_{OUT} = 0 \text{ mA}$ , Freq = $f_{max}$		140	mA

#### AUI INTERFACE PINS (TX $\pm$ , RX $\pm$ , and CD $\pm$ )

$V_{OD}$	Diff. Output Voltage (TX $\pm$ )	78 $\Omega$ Termination, and 270 $\Omega$ from Each to GND	$\pm 550$	$\pm 1200$	mV
$V_{OB}$	Diff. Output Voltage Imbalance (TX $\pm$ ) (Guaranteed by Design. Not Tested.)	78 $\Omega$ Termination, and 270 $\Omega$ from Each to GND	Typical: 40 mV		
$V_U$	Undershoot Voltage (TX $\pm$ ) (Guaranteed by Design. Not Tested.)	78 $\Omega$ Termination, and 270 $\Omega$ from Each to GND	Typical: 80 mV		
$V_{DS}$	Diff. Squelch Threshold (RX $\pm$ and CD $\pm$ )		-175	-300	mV

#### TPI INTERFACE PINS

$R_{TOL}$	TXOd $\pm$ , TXO $\pm$ Low Level Output Resistance	$I_{OL} = 25 \text{ mA}$		15	$\Omega$
$R_{TOH}$	TXOd $\pm$ , TXO $\pm$ High Level Output Resistance	$I_{OL} = -25 \text{ mA}$		15	$\Omega$
$V_{SRON1}$	Receive Threshold Turn-On Voltage 10BASE-T Mode	LOWSQL = 1	$\pm 300$	$\pm 585$	mV
$V_{SRON2}$	Receive Threshold Turn-On Voltage Reduce Threshold		$\pm 175$	$\pm 300$	mV
$V_{SROFF}$	Receive Threshold Turn-Off Voltage		$\pm 175$	$\pm 300$	mV
$V_{DIFF}$	Differential Mode Input Voltage Range	$V_{CC} = 5.0V$ (Guaranteed by Design, Not Tested)	-3.1	+3.1	V

#### OSCILLATOR PINS (OSCOU and OSCIN)

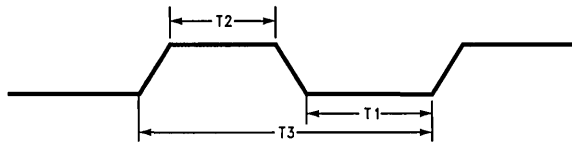
$V_{IH}$	OSCIN Input High Voltage	OSCIN is Connected to an Oscillator and OSCOUT is Open	2.0		V
$V_{IL}$	OSCIN Input Low Voltage	OSCIN is Connected to an Oscillator and OSCOUT is Open		0.8	V
$I_{OSC2}$	OSCIN Input Leakage Current	OSCIN is Connected to an Oscillator and OSCOUT is Open $V_{IN} = V_{CC}$ or GND	-100	100	$\mu\text{A}$



## 9.0 AC and DC Specifications (Continued)

### AC Characteristics

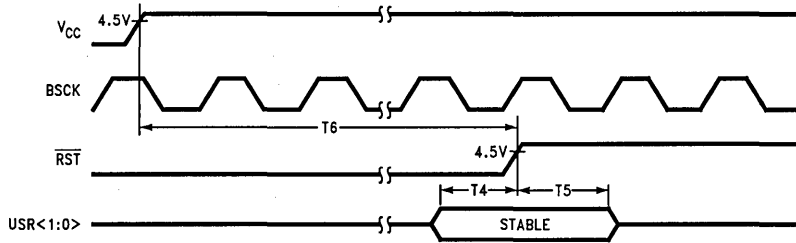
#### BUS CLOCK TIMING



TL/F/11719-57

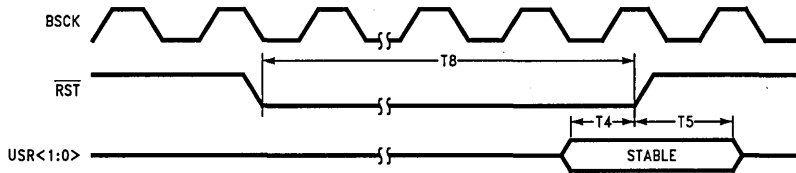
Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T1	Bus Clock Low Time	22		18		ns
T2	Bus Clock High Time	22		18		ns
T3	Bus Clock Cycle Time	50	100	40	100	ns

#### POWER-ON RESET



TL/F/11719-58

#### NON POWER-ON RESET



TL/F/11719-59

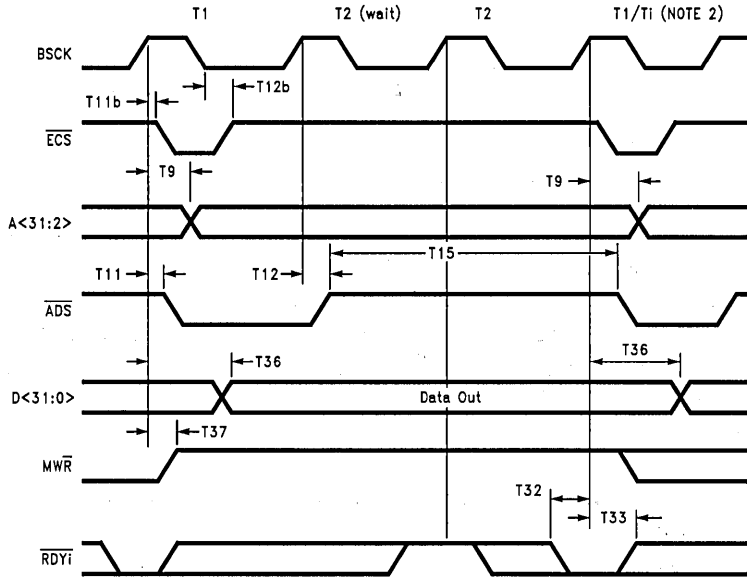
Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T4	USR<1:0> Setup to RST	7		6		ns
T5	USR<1:0> Hold from RST	9		8		ns
T6	Power-On Reset Low (Notes 1, 2)	10		10		TXC
T8	Reset Pulse Width (Notes 1, 2)	10		10		TXC

**Note 1:** The reset time is determined by the slower of BSCK or TXC. If BSCK > TXC, T6 and T8 equal 10 TXCs. If BSCK < TXC, T6 and T8 equal 10 BSCKs (T3).

**Note 2:** These specifications are not tested.

### 9.0 AC and DC Specifications (Continued)

MEMORY WRITE, BMODE = 0, SYNCHRONOUS MODE (one wait-state shown)



TL/F/11719-60

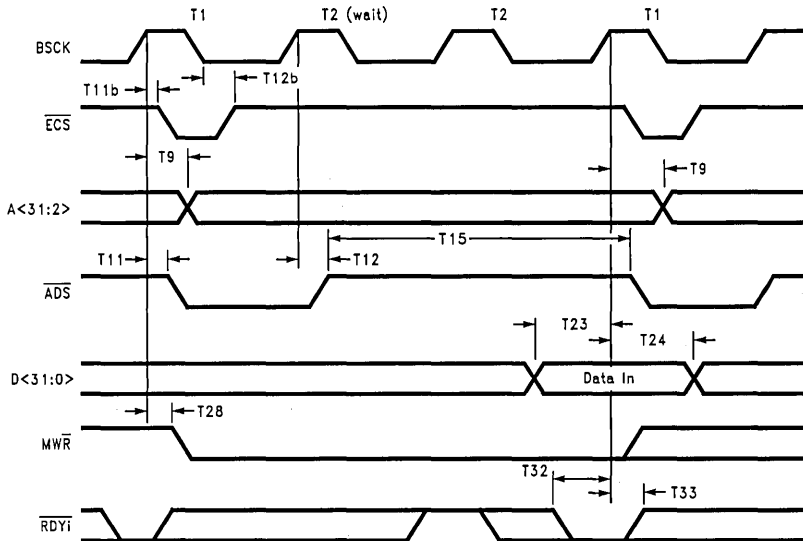
Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T9	BSCCK to Address Valid/Hold Time	2	26	3	24	ns
T11	BSCCK to $\overline{\text{ADS}}$ Low		26		24	ns
T11b	BSCCK to $\overline{\text{ECS}}$ Low		19		17	ns
T12	BSCCK to $\overline{\text{ADS}}$ High		24		22	ns
T12b	BSCCK to $\overline{\text{ECS}}$ High		29		27	ns
T15	$\overline{\text{ADS}}$ High Width	45		35		ns
T32	$\overline{\text{RDYi}}$ Setup to BSCCK	19		17		ns
T33	$\overline{\text{RDYi}}$ Hold from BSCCK	5		3		ns
T36	BSCCK to Memory Write Data Valid/Hold Time (Note 2)	3	50	3	48	ns
T37	BSCCK to $\overline{\text{MWR}}$ (Write) Valid (Note 1)		24		22	ns

**Note 1:** For successive read operations,  $\overline{\text{MWR}}$  remains high.

**Note 2:** One idle clock cycle (T1) will be inserted between the last write cycle and the following read cycle in RDA and TDA operation. Note that the data bus will become TRI-STATE from the rising edge of the clock after the idle cycle (see T52 for BSCCK to data TRI-STATE timing).

### 9.0 AC and DC Specifications (Continued)

MEMORY READ, BMODE = 0, SYNCHRONOUS MODE (one wait-state shown)



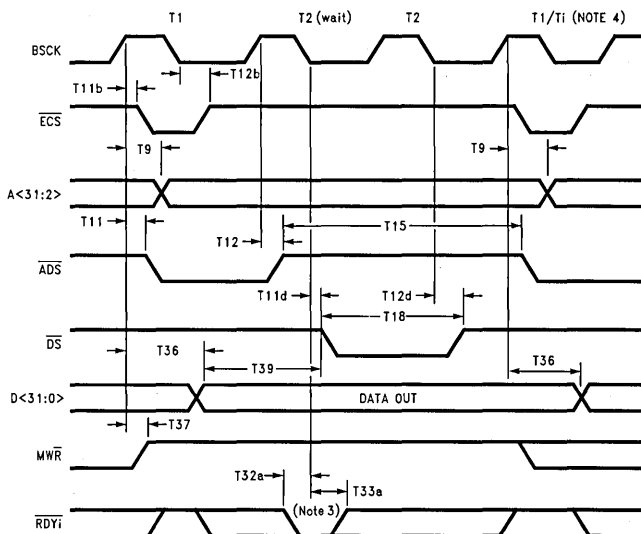
TL/F/11719-61

Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T9	BSCCK to Address Valid/Hold Time	3	26	3	24	ns
T11	BSCCK to $\overline{ADS}$ Low		26		24	ns
T11b	BSCCK to $\overline{ECS}$ Low		19		17	ns
T12	BSCCK to $\overline{ADS}$ High		24		22	ns
T12b	BSCCK to $\overline{ECS}$ High		29		27	ns
T15	$\overline{ADS}$ High Width	45		35		ns
T23	Read Data Setup Time to BSCCK	6		5		ns
T24	Read Data Hold Time from BSCCK	5		5		ns
T28	BSCCK to $\overline{MWR}$ (Ready) Valid (Note 1)		26		24	ns
T32	$\overline{RDYi}$ Setup Time to BSCCK	19		17		ns
T33	$\overline{RDYi}$ Hold Time to BSCCK	5		3		ns

Note 1: For successive read operations,  $\overline{MWR}$  remains low.

## 9.0 AC and DC Specifications (Continued)

### MEMORY WRITE, BMODE = 0, ASYNCHRONOUS MODE



TL/F/11719-62

Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T9	BSCCK to Address Valid/Hold Time	3	26	3	24	ns
T11	BSCCK to $\overline{\text{ADS}}$ Low		26		24	ns
T11b	BSCCK to $\overline{\text{ECS}}$ Low		19		17	ns
T11d	BSCCK to $\overline{\text{DS}}$ Low		17		15	ns
T12	BSCCK to $\overline{\text{ADS}}$ High		24		22	ns
T12b	BSCCK to $\overline{\text{ECS}}$ High		29		27	ns
T12d	BSCCK to $\overline{\text{DS}}$ High		17		15	ns
T15	$\overline{\text{ADS}}$ High Width	45		35		ns
T18	Write Data Strobe Low Width (Note 2)	40		30		ns
T32a	$\overline{\text{RDYi}}$ Asynchronous Setup to BSCCK (Note 3)	5		4		ns
T33a	$\overline{\text{RDYi}}$ Asynchronous Hold from BSCCK	5		5		ns
T36	BSCCK to Memory Write Data Valid/Hold Time (Note 4)	3	50	3	48	ns
T37	BSCCK to $\overline{\text{MWR}}$ (Write) Valid (Note 1)		24		22	ns
T39	Write Data Valid to $\overline{\text{DS}}$ Low	34		21		ns

**Note 1:** For successive read operations,  $\overline{\text{MWR}}$  remains high.

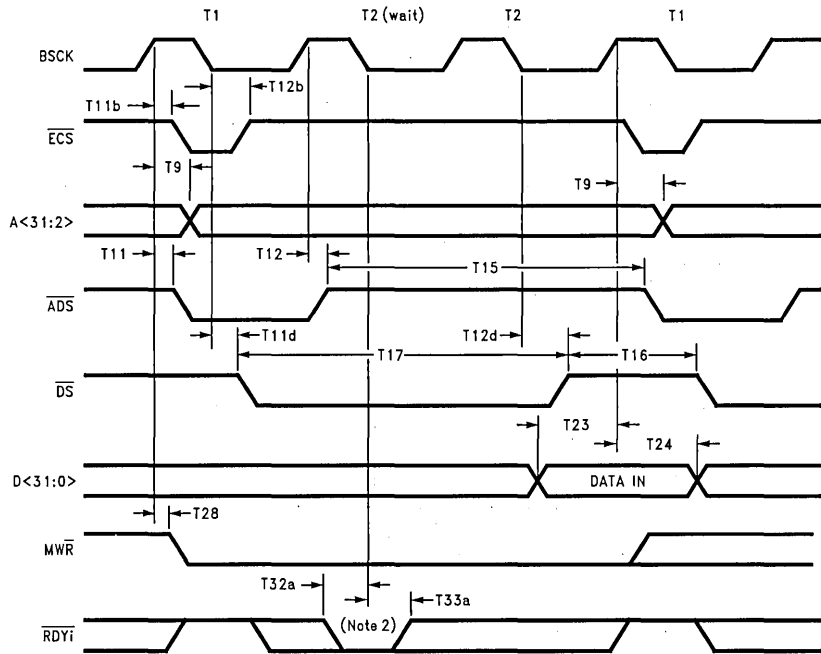
**Note 2:**  $\overline{\text{DS}}$  will only be asserted if the bus cycle has at least one wait state inserted.

**Note 3:** This setup time assures that the SONIC-T terminates the memory cycle on the next bus clock (BSCCK).  $\overline{\text{RDYi}}$  does not need to be synchronized to the bus clock, though, since it is an asynchronous input in this case.  $\overline{\text{RDYi}}$  is sampled during the falling edge of BSCCK. If the SONIC-T samples  $\overline{\text{RDYi}}$  low during the T1 cycle, the SONIC-T will finish the current access in a total of two bus clocks instead of three, which would be the case if  $\overline{\text{RDYi}}$  had been sampled low during T2 (wait). (This is assuming that programmable wait states are set to 0.)

**Note 4:** One idle clock cycle (Ti) will be inserted between the last write cycle and the following read cycle in RDA and TDA operation. Note that the data bus will become TRI-STATE from the rising edge of the clock after the idle cycle (see T52 for BSCCK to data TRI-STATE timing).

### 9.0 AC and DC Specifications (Continued)

MEMORY READ, BMODE = 0, ASYNCHRONOUS MODE



TL/F/11719-63

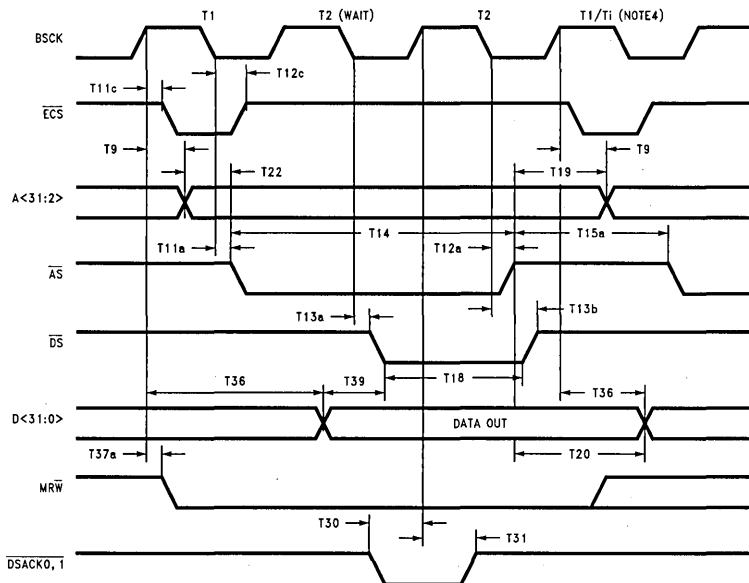
Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T9	BSCCK to Address Valid/Hold Time	3	26	3	24	ns
T11	BSCCK to $\overline{ADS}$ Low		26		24	ns
T11b	BSCCK to $\overline{ECS}$ Low		19		17	ns
T11d	BSCCK to $\overline{DS}$ Low		17		15	ns
T12	BSCCK to $\overline{ADS}$ High		24		22	ns
T12b	BSCCK to $\overline{ECS}$ High		29		27	ns
T12d	BSCCK to $\overline{DS}$ High		17		15	ns
T15	$\overline{ADS}$ High Width	45		35		ns
T16	Read Data Strobe High Width	45		35		ns
T17	Read Data Strobe Low Width	40		30		ns
T23	Read Data Setup Time to BSCCK	6		5		ns
T24	Read Data Hold Time from BSCCK	5		5		ns
T28	BSCCK to $\overline{MWR}$ (Read) Valid (Note 1)		26		24	ns
T32a	$\overline{RDYi}$ Asynchronous Setup Time to BSCCK (Note 2)	5		4		ns
T33a	$\overline{RDYi}$ Asynchronous Hold Time to BSCCK	5		5		ns

**Note 1:** For successive read operations,  $\overline{MWR}$  remains low.

**Note 2:** This setup time assures that the SONIC-T terminates the memory cycle on the next bus clock (BSCCK).  $\overline{RDYi}$  does not need to be synchronized to the bus clock, though, since it is an asynchronous input in this case.  $\overline{RDYi}$  is sampled during the falling edge of BSCCK. If the SONIC-T samples  $\overline{RDYi}$  low during the T1 cycle, the SONIC-T will finish the current access in a total of two bus clocks instead of three, which would be the case if  $\overline{RDYi}$  had been sampled low during T2 (wait). (This is assuming that programmable wait states are set to 0.)

## 9.0 AC and DC Specifications (Continued)

MEMORY WRITE, BMODE = 1, SYNCHRONOUS MODE (one wait-state shown)



TL/F/11719-64

Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T9	BSCCK to Address Valid/Hold Time	3	26	3	24	ns
T11a	BSCCK to $\overline{AS}$ Low		17		15	ns
T11c	BSCCK to $\overline{ECS}$ Low		19		17	ns
T12a	BSCCK to $\overline{AS}$ High		17		15	ns
T12c	BSCCK to $\overline{ECS}$ High		19		17	ns
T13a	BSCCK to $\overline{DS}$ High (Note 1)		16		14	ns
T13b	BSCCK to $\overline{DS}$ High (Note 1)		16		14	ns
T14	$\overline{AS}$ Low Width	44		34		ns
T15a	$\overline{AS}$ High Width	45		35		ns
T18	Write Data Strobe Width (Note 1)	40		30		ns
T19	Address Hold Time from $\overline{AS}$	18		14		ns
T20	Data Hold Time from $\overline{AS}$	20		16		ns
T22	Address Valid to $\overline{AS}$ (Note 3)	9		6		ns
T30	$\overline{DSACK0,T}$ Setup to BSCCK (Note 3)	5		4		ns
T31	$\overline{DSACK0,T}$ Hold from BSCCK	9		8		ns
T36	BSCCK to Memory Write Data Valid/Hold Time (Note 4)	3	50	3	48	ns
T37a	BSCCK to $\overline{MRW}$ (Write) Valid (Note 2)		26		24	ns
T39	Write Data Valid to Data Strobe Low	34		21		ns

Note 1:  $\overline{DS}$  will only be asserted if the bus cycle has at least one wait state inserted.

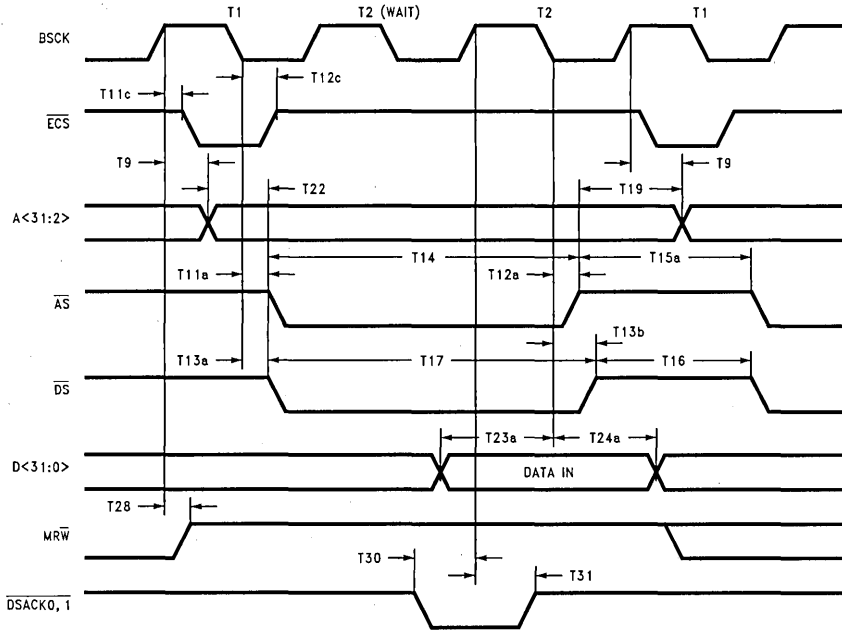
Note 2: For successive write operations,  $\overline{MRW}$  remains low.

Note 3:  $\overline{DSACK0,T}$  must be synchronized to the bus clock (BSCCK) during synchronous mode.

Note 4: One idle clock cycle (T<sub>i</sub>) will be inserted between the last write cycle and the following read cycle in RDA and TDA operation. Note that the data bus will become TRI-STATE from the rising edge of the clock after the idle cycle (see T52 for BSCCK to data TRI-STATE timing).

### 9.0 AC and DC Specifications (Continued)

MEMORY READ, BMODE = 1, SYNCHRONOUS MODE (one wait-state shown)



TL/F/11719-65

Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T9	BSCCK to Address Valid	3	26	3	24	ns
T11a	BSCCK to $\overline{AS}$ Low		17		15	ns
T11c	BSCCK to $\overline{ECS}$ Low		19		17	ns
T12a	BSCCK to $\overline{AS}$ High		17		15	ns
T12c	BSCCK to $\overline{ECS}$ High		19		17	ns
T13a	BSCCK to $\overline{DS}$ Low (Note 3)		16		14	ns
T13b	BSCCK to $\overline{DS}$ High (Note 3)		16		14	ns
T14	$\overline{AS}$ Low Width	44		34		ns
T15a	$\overline{AS}$ High Width	45		35		ns
T16	Read Data Strobe High Width	45		35		ns
T17	Read Data Strobe Low Width	40		30		ns
T19	Address Hold Time from $\overline{AS}$	18		14		ns
T22	Address Valid to $\overline{AS}$	9		6		ns
T23a	Read Data Setup Time to BSCCK	5		4		ns
T24a	Read Data Hold Time from BSCCK	5		5		ns
T28	BSCCK to $\overline{MRW}$ (Read) Valid (Note 1)		26		24	ns
T30	$\overline{DSACK0,1}$ Setup to BSCCK (Note 2)	5		4		ns
T31	$\overline{DSACK0,1}$ Hold from BSCCK	9		8		ns

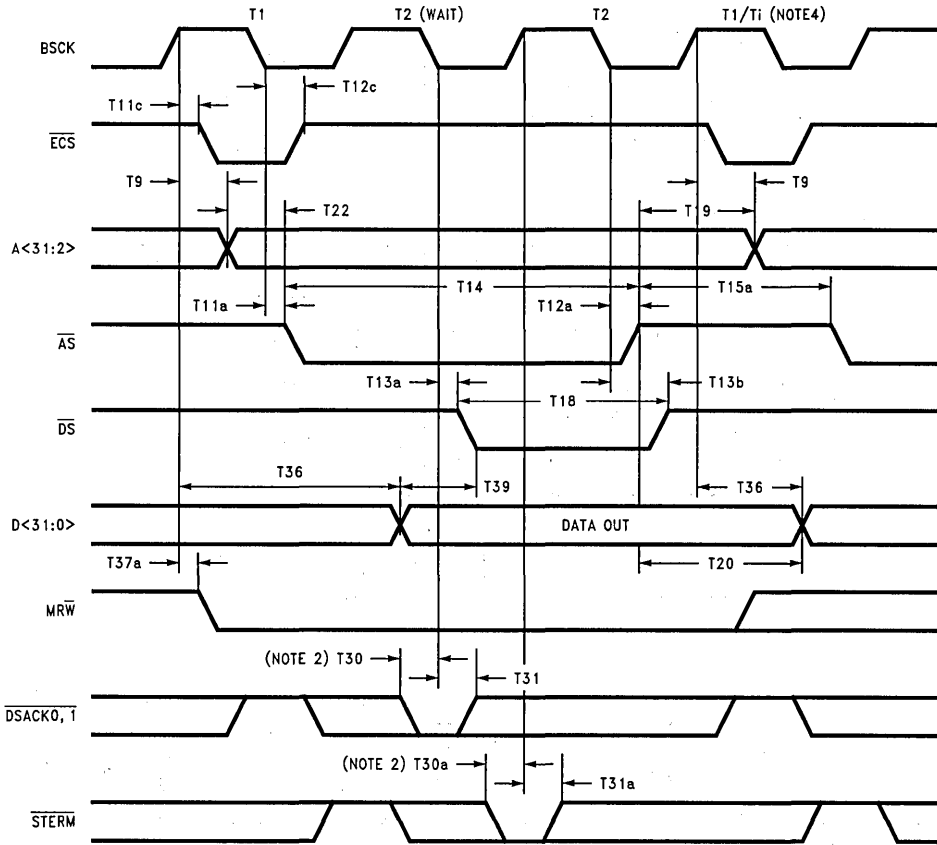
Note 1: For successive read operations,  $\overline{MRW}$  remains high.

Note 2:  $\overline{DSACK0,1}$  must be synchronized to the bus clock (BSCCK) during synchronous mode.

Note 3:  $\overline{DS}$  will only be asserted if the bus cycle has at least one wait state inserted.

### 9.0 AC and DC Specifications (Continued)

#### MEMORY WRITE, BMODE = 1, ASYNCHRONOUS MODE



TL/F/11719-66



## 9.0 AC and DC Specifications (Continued)

Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T9	BCK to Address Valid	3	26	3	24	ns
T11a	BCK to $\overline{AS}$ Low		17		15	ns
T11c	BCK to $\overline{ECS}$ Low		19		17	ns
T12a	BCK to $\overline{AS}$ High		17		15	ns
T12c	BCK to $\overline{ECS}$ High		19		17	ns
T13a	BCK to $\overline{DS}$ Low		16		14	ns
T13b	BCK to $\overline{DS}$ High		16		14	ns
T14	$\overline{AS}$ Low Width	44		34		ns
T15a	$\overline{AS}$ High Width	45		35		ns
T18	Write Data Strobe Low Width (Note 3)	40		30		ns
T19	Address Hold Time from $\overline{AS}$	18		14		ns
T20	Data Hold Time from $\overline{AS}$	20		16		ns
T22	Address Valid to $\overline{AS}$	9		6		ns
T30	$\overline{DSACK0,1}$ Setup to BCK (Note 2)	5		4		ns
T30a	$\overline{STERM}$ Setup to BCK (Note 2)	5		4		ns
T31	$\overline{DSACK0,1}$ Hold from BCK	9		8		ns
T31a	$\overline{STERM}$ Hold from BCK	8		7		ns
T36	BCK to Memory Write Data Valid (Note 4)	3	50	3	48	ns
T37a	BCK to $\overline{MRW}$ (Write) Valid (Note 1)		26		24	ns
T39	Write Data Valid to Data Strobe Low	34		21		ns

**Note 1:** For successive write operations,  $\overline{MRW}$  remains low.

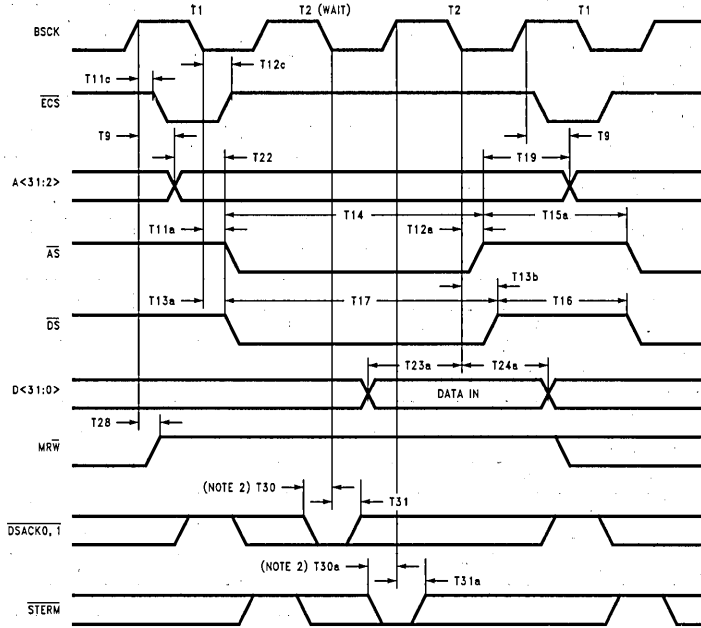
**Note 2:** Meeting the setup time for  $\overline{DSACK0,1}$  or  $\overline{STERM}$  guarantees that the SONIC-T will terminate the memory cycle 11/2 bus clocks after  $\overline{DSACK0,1}$  were sampled, or 1 cycle after  $\overline{STERM}$  was sampled. T2 states will be repeated until  $\overline{DSACK0,1}$  or  $\overline{STERM}$  are sampled properly in a low state. If the SONIC-T samples  $\overline{DSACK0,1}$  or  $\overline{STERM}$  low during the T1 or first T2 state respectively, the SONIC-T will finish the current access in a total of two bus clocks instead of three (assuming that programmable wait states are set to 0).  $\overline{DSACK0,1}$  are asynchronously sampled and  $\overline{STERM}$  is synchronously sampled.

**Note 3:**  $\overline{DS}$  will only be asserted if the bus cycle has at least one wait state inserted.

**Note 4:** One idle clock cycle (T1) will be inserted between the last write cycle and the following read cycle in RDA and TDA operation. Note that the data bus will become TRI-STATE from the rising edge of the clock after the idle cycle (see T52 for BCK to data TRI-STATE timing).

## 9.0 AC and DC Specifications (Continued)

### MEMORY READ, BMODE = 1, ASYNCHRONOUS MODE



TL/F/11719-67

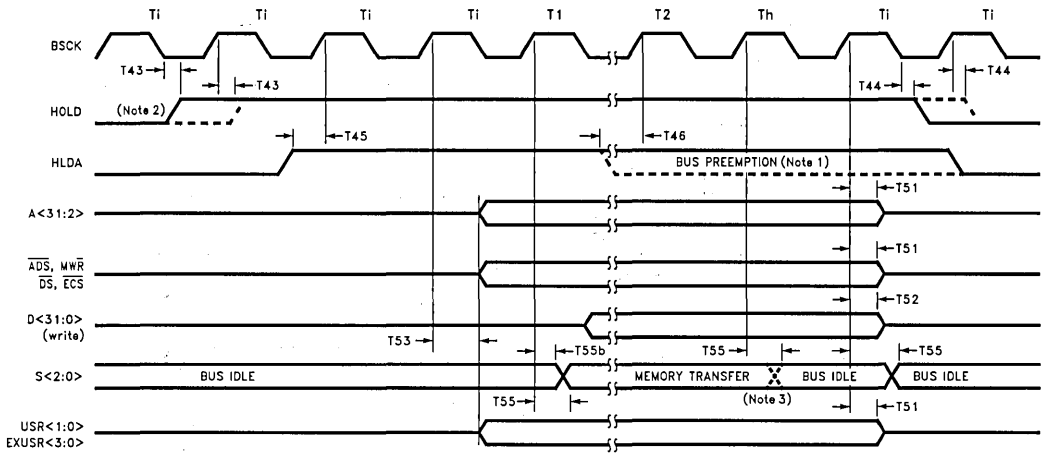
Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T9	BSCK to Address Valid	3	26	3	24	ns
T11a	BSCK to $\overline{AS}$ Low		17		15	ns
T11c	BSCK to $\overline{ECS}$ Low		19		17	ns
T12a	BSCK to $\overline{AS}$ High		17		15	ns
T12c	BSCK to $\overline{ECS}$ High		19		17	ns
T13a	BSCK to $\overline{DS}$ Low		16		14	ns
T13b	BSCK to $\overline{DS}$ High		16		14	ns
T14	$\overline{AS}$ Low Width	44		34		ns
T15a	$\overline{AS}$ High Width	45		35		ns
T16	Read Data Strobe High Width	45		35		ns
T17	Read Data Strobe Low Width	40		30		ns
T19	Address Hold Time from $\overline{AS}$	18		14		ns
T22	Address Valid to $\overline{AS}$	9		6		ns
T23a	Read Data Setup Time to BSCK	6		5		ns
T24a	Read Data Hold Time from BSCK	5		5		ns
T28	BSCK to $\overline{MRW}$ (Read) Valid (Note 1)		26		24	ns
T30	$\overline{DSACK0,1}$ Setup to BSCK (Note 2)	5		4		ns
T30a	$\overline{STERM}$ Setup to BSCK (Note 2)	5		4		ns
T31	$\overline{DSACK0,1}$ Hold from BSCK	9		8		ns
T31a	$\overline{STERM}$ Hold from BSCK	8		7		ns

**Note 1:** For successive read operations,  $\overline{MRW}$  remains high.

**Note 2:** Meeting the setup time for  $\overline{DSACK0,1}$  or  $\overline{STERM}$  guarantees that the SONIC-T will terminate the memory cycle 1.5 bus clocks after  $\overline{DSACK0,1}$  were sampled, or 1 cycle after  $\overline{STERM}$  was sampled. T2 states will be repeated until  $\overline{DSACK0,1}$  or  $\overline{STERM}$  are sampled properly in a low state. If the SONIC-T samples  $\overline{DSACK0,1}$  or  $\overline{STERM}$  low during the T1 or first T2 state respectively, the SONIC-T will finish the current access in a total of two bus clocks instead of three (assuming that programmable wait states are set to 0).  $\overline{DSACK0,1}$  are asynchronously sampled and  $\overline{STERM}$  is synchronously sampled.

## 9.0 AC and DC Specifications (Continued)

### BUS REQUEST TIMING, BMODE = 0



TL/F/11719-68

Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T43	BSCCK to HOLD High (Note 2)		18	16		ns
T44	BSCCK to HOLD Low (Note 2)		19	17		ns
T45	HLDA Asynchronous Setup Time to BSCCK	7		6		ns
T46	HLDA Synchronous Deassert Setup Time (Note 1)	7		6		ns
T51	BSCCK to Address, $\overline{ADS}$ , $\overline{MWR}$ , $\overline{DS}$ , $\overline{ECS}$ , $USR<1:0>$ and $EXUSR<3:0>$ TRI-STATE (Note 4)		37	35		ns
T52	BSCCK to Data TRI-STATE		39	37		ns
T53	BSCCK to $USR<1:0>$ or $EXUSR<3:0>$ Valid		34	32		ns
T55	BSCCK to Bus Status Valid		29	27		ns
T55b	$S<2:0>$ Hold from BSCCK	3		3		ns

**Note 1:** A block transfer by the SONIC-T can be pre-empted from the bus by deasserting HLDA provided HLDA is asserted T46 before the rising edge of the last T2 in the current access.

**Note 2:** The assertion edge for HOLD is dependent upon the PH bit in the DCR2. The default situation is shown with a solid line in the timing diagram. T43 and T44 apply for both modes. Also, if HLDA is asserted when the SONIC-T wants to acquire the bus, HOLD will not be asserted until HLDA has been deasserted first.

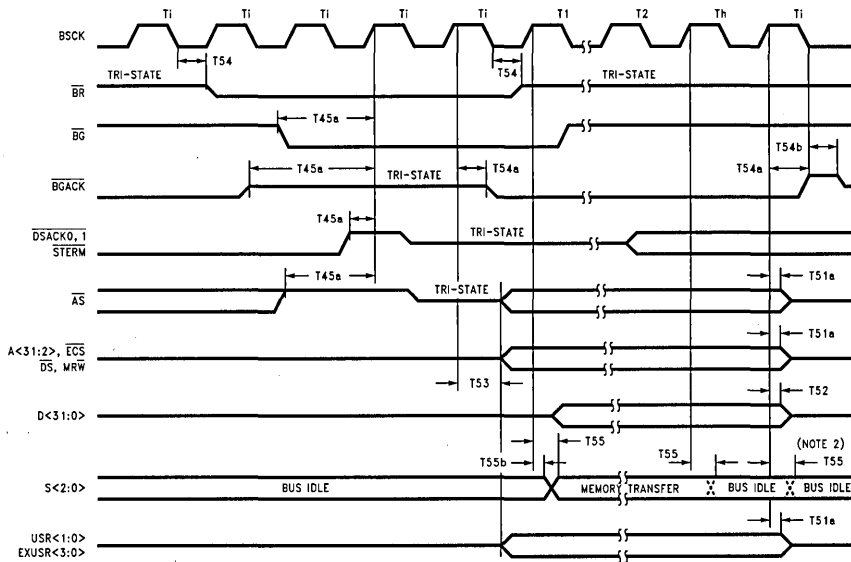
**Note 3:**  $S<2:0>$  will indicate IDLE at the end of T2 if the last operation is a read operation, or at the end of Th if the last operation is a write operation.

**Note 4:** This timing value includes an RC delay inherent in the test measurement. These signals typically TRI-STATE 7 ns earlier, enabling other devices to drive these lines without contention.



## 9.0 AC and DC Specifications (Continued)

### BUS REQUEST TIMING, BMODE = 1



TL/F/11719-69

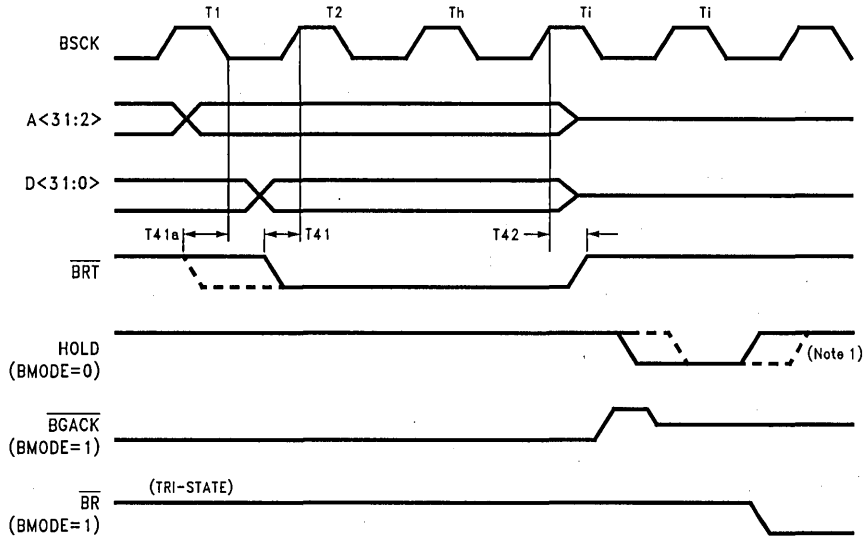
Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T45a	$\overline{BG}$ , $\overline{AS}$ , $\overline{BGACK}$ , $\overline{DSACK0,1}$ , and $\overline{STERM}$ Asynchronous Setup Time to BSCK (Note 1)	7		6		ns
T51a	BSCK to Address, $\overline{AS}$ , $\overline{MRW}$ , $\overline{DS}$ , $\overline{ECS}$ , $USR<1:0>$ and $EXUSR<3:0>$ TRI-STATE		37		35	ns
T52	BSCK to Data TRI-STATE		34		32	ns
T53	BSCK to Address, $\overline{AS}$ , $\overline{MRW}$ , $\overline{DS}$ , $\overline{ECS}$ , $USR<1:0>$ and $EXUSR<3:0>$ Active (Note 1)		34		32	ns
T54	BSCK Low to $\overline{BR}$ Low/TRI-STATE		26		24	ns
T54a	BSCK High to $\overline{BGACK}$ Low/High		24		22	ns
T54b	High to $\overline{BGACK}$ TRI-STATE		19		17	ns
T55	BSCK to Bus Status Valid		29		27	ns
T55b	$S<2:0>$ Hold from BSCK	3		3		ns

**Note 1:**  $\overline{BGACK}$  is asserted one bus clock after all the signals ( $\overline{AS}$ ,  $\overline{DSACK0,1}$ ,  $\overline{BGACK}$ ,  $\overline{STERM}$  (Extended bus mode), and  $\overline{BG}$ ) meet the T45a setup time (see Section 5.4.1 for more information). The address bus,  $\overline{AS}$ ,  $\overline{DS}$ ,  $\overline{ECS}$ ,  $\overline{MRW}$ ,  $USR<1:0>$ , and  $EXUSR<3:0>$  will also be driven active on the same clock.

**Note 2:**  $S<2:0>$  will indicate IDLE at the end of T2 if the last operation is a read operation, or at the end of Th if the last operation is a write operation.

## 9.0 AC and DC Specifications (Continued)

### BUS RETRY



TL/F/11719-70

Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T41	Bus Retry Synchronous Setup Time to BSCk (Note 3)	5		4		ns
T41a	Bus Retry Asynchronous Setup Time to BSCk (Note 3)	6		5		ns
T42	Bus Retry Hold Time from BSCk (Note 2)	7		6		ns

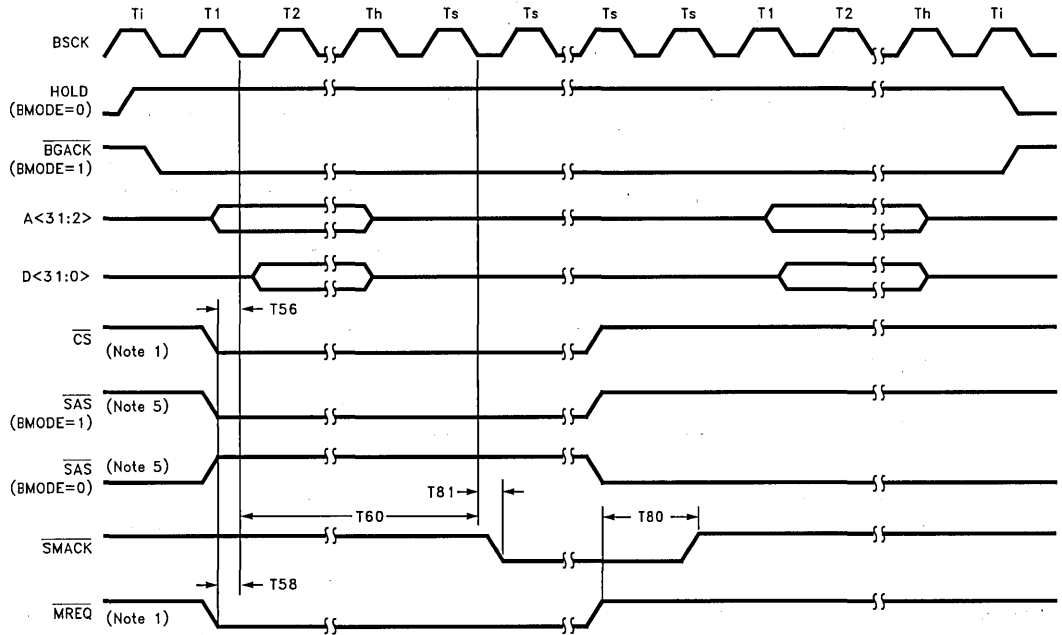
**Note 1:** Depending upon the mode, the SONIC-T will assert and deassert HOLD from the rising or falling edge of BSCk.

**Note 2:** Unless Latched Bus Retry mode is set (LBR in the Data Configuration Register, Section 4.3.2), BRT must remain asserted until after the Th state. If Latched Bus Retry mode is used, BRT does not need to satisfy T42.

**Note 3:** T41 is for synchronous bus retry and T41a is for asynchronous bus retry (see Section 4.3.2, bit 15, Extended Bus Mode). Since T41a is an asynchronous setup time, it is not necessary to meet it, but doing so will guarantee that the bus exception occurs in the current memory transfer, not the next.

## 9.0 AC and DC Specifications (Continued)

### MEMORY ARBITRATION/S�AVE ACCESS



TL/F/11719-71

Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T56	$\overline{CS}$ Low Asynchronous Setup to BSCCK (Note 2)	8		7		ns
T58	$\overline{MREQ}$ Low Asynchronous Setup to BSCCK (Note 2)	8		7		ns
T60	$\overline{MREQ}$ or $\overline{CS}$ Valid to $\overline{SMACK}$ Low (Notes 3, 4)	1	5	1	5	bcyc
T80	$\overline{MREQ}$ to $\overline{SMACK}$ High		18		16	ns
T81	BSCCK to $\overline{SMACK}$ Low		22		20	ns

**Note 1:** Both  $\overline{CS}$  and  $\overline{MREQ}$  must not be asserted concurrently. If these signals are successively asserted, there must be at least two bus clocks between the deasserting and asserting edges of these signals.

**Note 2:** It is not necessary to meet the setup times for  $\overline{MREQ}$  or  $\overline{CS}$  since these signals are asynchronously sampled. Meeting the setup time for these signals, however, makes it possible to use T60 to determine exactly when  $\overline{SMACK}$  will be asserted.

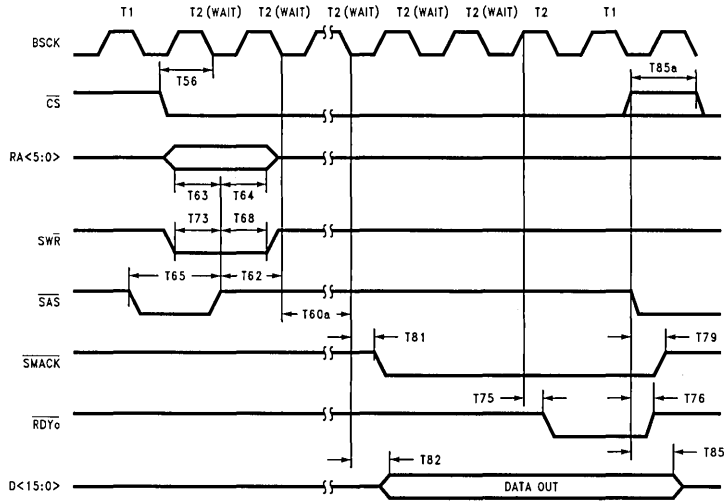
**Note 3:** T60 could range from 1 bus clock minimum to 5 bus clock maximum depending on what state machine the SONIC-T is when the  $\overline{CS}$  or  $\overline{MREQ}$  signal is asserted. This timing is not tested, but is guaranteed by design. This specification assumes that  $\overline{CS}$  or  $\overline{MREQ}$  is asserted before the falling edge that these signals are asynchronously clocked in on (see T56 and T58).  $\overline{SAS}$  must have been asserted for this timing to be correct. See  $\overline{SAS}$  and  $\overline{CS}$  timing in the Register Read, and Register Write timing specifications.

**Note 4:** bcyc = bus clock cycle time (T3).

**Note 5:** The way in which  $\overline{SMACK}$  is asserted due to  $\overline{CS}$  is not the same as the way in which  $\overline{SMACK}$  is asserted due to  $\overline{MREQ}$ .  $\overline{SMACK}$  goes low as a direct result of the assertion of  $\overline{MREQ}$ , whereas, for  $\overline{CS}$ ,  $\overline{SAS}$  must also be driven low (BMODE = 1) or high (BMODE = 0) before  $\overline{SMACK}$  will be asserted. This means that when  $\overline{SMACK}$  is asserted due to  $\overline{MREQ}$ ,  $\overline{SMACK}$  will remain asserted until  $\overline{MREQ}$  is deasserted. Multiple memory accesses can be made to the shared memory without  $\overline{SMACK}$  ever going high. When  $\overline{SMACK}$  is asserted due to  $\overline{CS}$ , however,  $\overline{SMACK}$  will only remain low as long as  $\overline{SAS}$  is also low (BMODE = 1) or high (BMODE = 0).  $\overline{SMACK}$  will not remain low throughout multiple register accesses to the SONIC-T because  $\overline{SAS}$  must toggle for each register access. This is an important difference to consider when designing shared memory designs.

## 9.0 AC and DC Specifications (Continued)

### REGISTER READ, BMODE = 0 (Note 1)



TL/F/11719-88

Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T56	$\overline{CS}$ Asynchronous Setup to BSCk (Notes 4, 6)	8		7		ns
T60a	$\overline{CS}$ and $\overline{SAS}$ to $\overline{SMACK}$ Low (Notes 3, 5, 6)	0	4	0	4	bcyc
T62	$\overline{SAS}$ Asynchronous Setup to BSCk (Notes 4, 6)	7		6		ns
T63	Register Address Setup Time to $\overline{SAS}$	7		6		ns
T64	Register Address Hold Time from $\overline{SAS}$	8		7		ns
T65	$\overline{SAS}$ Minimum Low Width (Notes 4, 6)	20		17		ns
T68	$\overline{SWR}$ (Read) Hold from $\overline{SAS}$	8		7		ns
T73	$\overline{SWR}$ (Read) Setup to $\overline{SAS}$	7		6		ns
T75	BSCk to $\overline{RDY0}$ Low		20		18	ns
T76	$\overline{SAS}$ or $\overline{CS}$ to $\overline{RDY0}$ High (Note 2)		34		32	ns
T79	$\overline{SAS}$ or $\overline{CS}$ to $\overline{SMACK}$ High (Note 2)		18		16	ns
T81	BSCk to $\overline{SMACK}$ Low		22		20	ns
T82	BSCk to Register Data Valid		44		42	ns
T85	$\overline{SAS}$ or $\overline{CS}$ to Data TRI-STATE (Notes 2, 7)		34		32	ns
T85a	Minimum $\overline{CS}$ Deassert Time (Note 3)	1		1		bcyc

**Note 1:** This figure shows a slave access to the SONIC-T. The BSCk states (T1, T2, etc.) are the equivalent processor states during a slave access.

**Note 2:** If  $\overline{CS}$  is deasserted before the falling edge of  $\overline{SAS}$ , T76, T79 and T85 are referenced from the rising edge of  $\overline{CS}$ .

**Note 3:** bcyc = bus clock cycle time (T3).

**Note 4:** It is not necessary to meet the setup time for  $\overline{CS}$  (T56) and the setup time for  $\overline{SAS}$  (T62) since these signals are asynchronously sampled. Meeting these setup times for these signals, however, makes it possible to use T60a to determine exactly when  $\overline{SMACK}$  will be asserted. For multiple register accesses,  $\overline{CS}$  can be held low and  $\overline{SAS}$  can be used to delimit the slave cycle. In this case,  $\overline{SMACK}$  will be driven low by the SONIC-T after T60a when T62 is met. T85a must be met to ensure proper slave operation once  $\overline{CS}$  is deasserted.

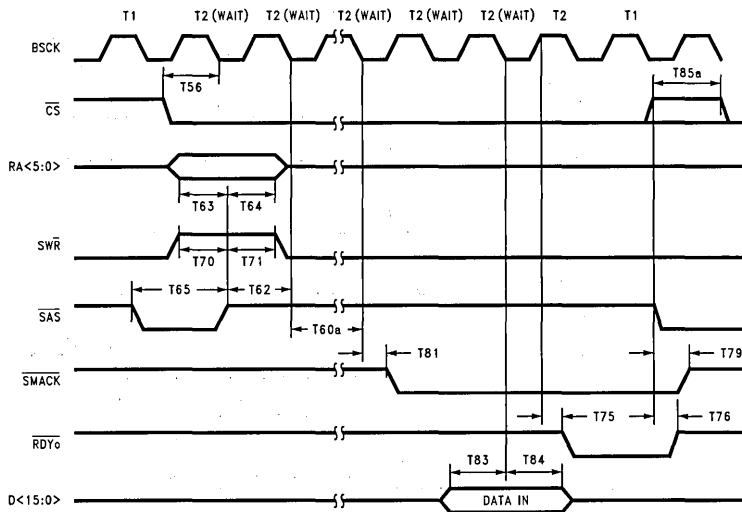
**Note 5:** The smaller value for T60a refers to when the SONIC-T is accessed during an Idle condition and the other value refers to when the SONIC-T is accessed during non-idle conditions. These values are not tested, but are guaranteed by design.

**Note 6:**  $\overline{SAS}$  may be asserted low anytime before or simultaneous to the falling edge of  $\overline{CS}$ . Register address and slave read/write signals are latched on the rising edge of the  $\overline{SAS}$ , and if T62 is met,  $\overline{SMACK}$  will be asserted by the SONIC-T after T60a. If T62 is not met, SONIC-T will sample  $\overline{SAS}$  again on the next falling edge of the clock, and  $\overline{SMACK}$  will not be asserted until  $\overline{SAS}$  is deasserted.

**Note 7:** This timing value includes an RC delay inherent in the test measurement. These signals typically TRI-STATE 7 ns earlier, enabling other devices to drive these lines without contention.

## 9.0 AC and DC Specifications (Continued)

### REGISTER WRITE, BMODE = 0 (Note 1)



TL/F/11719-89

Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T56	$\overline{CS}$ Asynchronous Setup to BSCCK (Notes 4, 6)	8		7		ns
T60a	$\overline{CS}$ and $\overline{SAS}$ to $\overline{SMACK}$ Low (Notes 3, 5, 6)	0	4	0	4	bcyc
T62	$\overline{SAS}$ Asynchronous Setup to BSCCK (Notes 4, 6)	7		6		ns
T63	Register Address Setup Time to $\overline{SAS}$	7		6		ns
T64	Register Address Hold Time from $\overline{SAS}$	8		7		ns
T65	Minimum $\overline{SAS}$ Low Width (Notes 4, 6)	20		17		ns
T70	$\overline{SWR}$ (Write) Setup to $\overline{SAS}$	7		6		ns
T71	$\overline{SWR}$ (Write) Hold from $\overline{SAS}$	8		7		ns
T75	BSCCK to $\overline{RDY<0>}$ Low		20		18	ns
T76	$\overline{SAS}$ or $\overline{CS}$ to $\overline{RDY<0>}$ High (Note 2)		34		32	ns
T79	$\overline{SAS}$ or $\overline{CS}$ to $\overline{SMACK}$ High (Note 2)		18		16	ns
T81	BSCCK to $\overline{SMACK}$ Low		22		20	ns
T83	Register Write Data Setup to BSCCK	8		7		ns
T84	Register Write Data Hold from BSCCK	14		12		ns
T85a	Minimum $\overline{CS}$ Deassert Time (Note 3)	1		1		bcyc

**Note 1:** This figure shows a slave access to the SONIC-T. The BSCCK states (T1, T2, etc.) are the equivalent processor states during a slave access.

**Note 2:** If  $\overline{CS}$  is deasserted before the falling edge of  $\overline{SAS}$ , T76, T79 and T85 are referenced from the rising edge of  $\overline{CS}$ .

**Note 3:** bcyc = bus clock cycle time (T3).

**Note 4:** It is not necessary to meet the setup time for  $\overline{CS}$  (T56) and the setup time for  $\overline{SAS}$  (T62) since these signals are asynchronously sampled. Meeting these setup times for these signals, however, makes it possible to use T60a to determine exactly when  $\overline{SMACK}$  will be asserted. For multiple register accesses,  $\overline{CS}$  can be held low and  $\overline{SAS}$  can be used to delimit the slave cycle. In this case,  $\overline{SMACK}$  will be driven low by the SONIC-T after T60a when T62 is met. T85a must be met to ensure proper slave operation once  $\overline{CS}$  is deasserted.

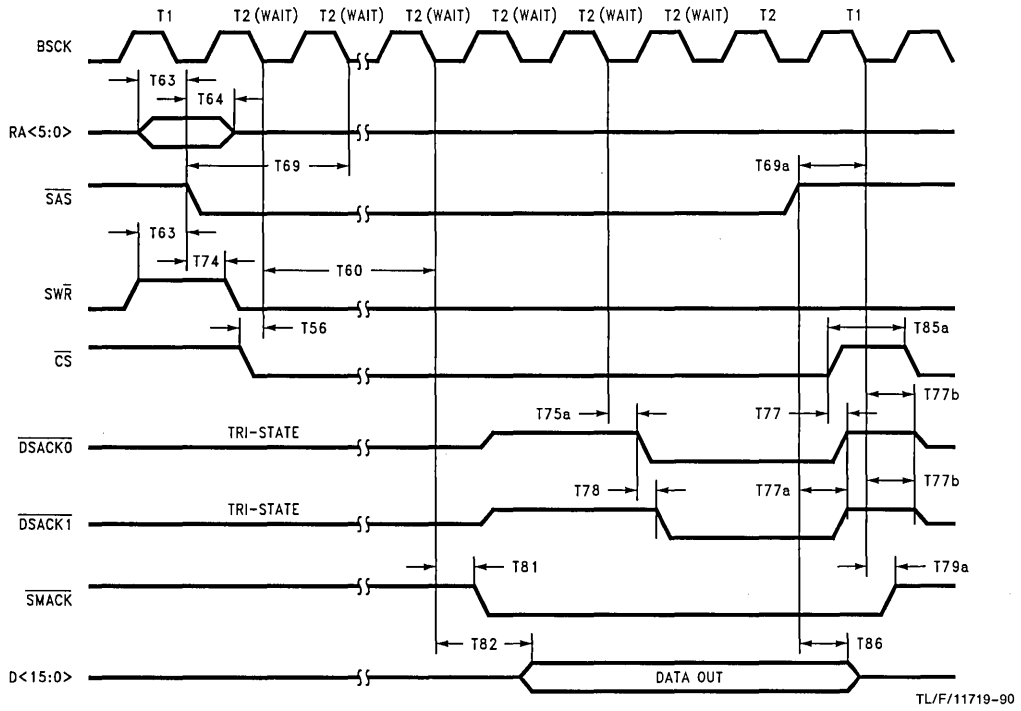
**Note 5:** The smaller value for T60a refers to when the SONIC-T is accessed during an Idle condition and the other value refers to when the SONIC-T is accessed during non-idle conditions. These values are not tested, but are guaranteed by design.

**Note 6:**  $\overline{SAS}$  may be asserted low anytime before or simultaneous to the falling edge of  $\overline{CS}$ . Register address and slave read/write signals are latched on the rising edge of the  $\overline{SAS}$ , and if T62 is met,  $\overline{SMACK}$  will be asserted by the SONIC-T after T60a. If T62 is not met, SONIC-T will sample  $\overline{SAS}$  again on the next falling edge of the clock, and  $\overline{SMACK}$  will not be asserted until  $\overline{SAS}$  is deasserted.



### 9.0 AC and DC Specifications (Continued)

REGISTER READ, BMODE = 1 (Note 1)



## 9.0 AC and DC Specifications (Continued)

Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T56	$\overline{CS}$ Asynchronous Setup to BSCK (Notes 3, 4)	8		7		ns
T60	$\overline{CS}$ Valid to $\overline{SMACK}$ Low (Notes 2, 3, 4)	1	5	1	5	bcyc
T63	Register Address Setup to $\overline{SAS}$	6		5		ns
T64	Register Address Hold from $\overline{SAS}$	8		7		ns
T67	SRW (Read) Setup to $\overline{SAS}$	4		3		ns
T69	$\overline{SAS}$ Asynchronous Setup to BSCK (Notes 3, 4)	7		6		ns
T69a	$\overline{SAS}$ Asynchronous Setup to BSCK (Notes 3, 5)	5		4		ns
T74	SRW (Read) Hold from $\overline{SAS}$	8		7		ns
T75a	BSCK to $\overline{DSACK0,1}$ Low		20		18	ns
T77	$\overline{CS}$ to $\overline{DSACK0,1}$ High (Note 5)		20		18	ns
T77a	$\overline{SAS}$ to $\overline{DSACK0,1}$ High (Note 5)		31		29	ns
T77b	BSCK to $\overline{DSACK0,1}$ TRI-STATE (Note 5)		19		17	ns
T78	Skew between $\overline{DSACK0,1}$		3		3	ns
T79a	BSCK to $\overline{SMACK}$ High (Note 5)		19		17	ns
T81	BSCK to $\overline{SMACK}$ Low		22		20	ns
T82	BSCK to Register Data Valid		44		42	ns
T85a	Minimum $\overline{CS}$ Deassert Time (Notes 2, 3)	1		1		bcyc
T86	$\overline{SAS}$ to Register Data TRI-STATE (Note 6)		42		40	ns

**Note 1:** This figure shows a slave access to the SONIC-T when the SONIC-T is idle, or rather not in master mode. If the SONIC-T is a bus master, there will be some differences as noted in the Memory Arbitration/Slave Access diagram. The BSCK states (T1, T2, etc.) are the equivalent processor states during a slave access.

**Note 2:** bcyc = bus clock cycle time (T3).

**Note 3:** It is not necessary to meet the setup time for  $\overline{CS}$  and  $\overline{SAS}$  (T56 and T69) since these signals are asynchronously sampled. Meeting the setup time for these signals, however, makes it possible to use T60 to determine when  $\overline{SMACK}$  will be asserted.  $\overline{SAS}$  may be asserted anytime before the next falling edge of the clock that the  $\overline{CS}$  is sampled on (as shown by specification T69). For multiple register accesses,  $\overline{CS}$  can be held low and  $\overline{SAS}$  can be used to delimit the slave cycle (T69a must be met in order to terminate and start another cycle). In this case,  $\overline{SMACK}$  will be asserted as soon as T69 timing is met.

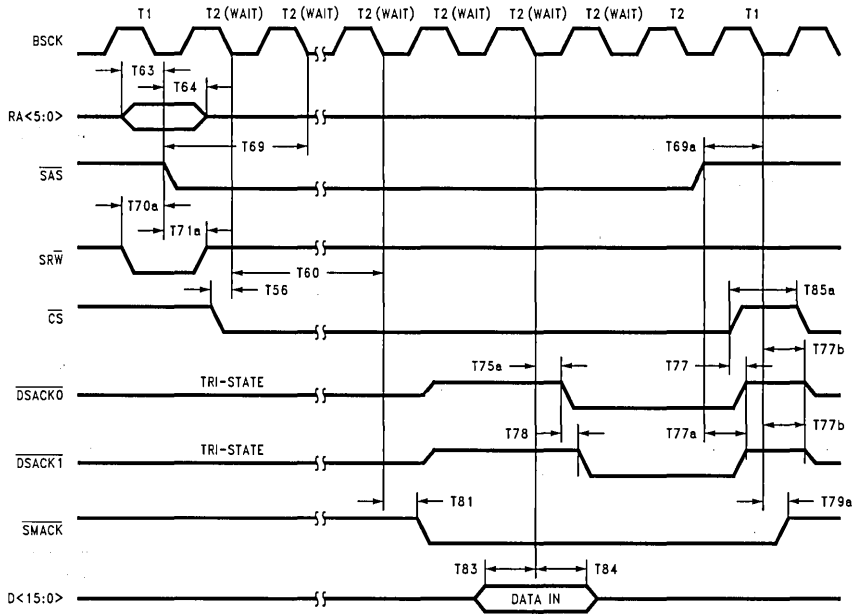
**Note 4:** T60 could range from 1 bus clock minimum to 5 bus clock maximum depending on what state machine the SONIC-T is in when the  $\overline{CS}$  signal is asserted. This timing is not tested, but is guaranteed by design. This specification assumes that both T56 is met for  $\overline{CS}$  and T69 is met for  $\overline{SAS}$ . T60 specification also assumes that there were no wait states in the current master mode access (if  $\overline{CS}$  is asserted when SONIC-T is in Master Mode). If there were wait states, then it would increase to T60 further.

**Note 5:** It is not necessary to meet the setup time for  $\overline{SAS}$  (T69a) since this signal is asynchronously sampled. Meeting the setup time for this signal, however, will ensure  $\overline{DSACK0,1}$  becomes TRI-STATE (T77b) and  $\overline{SMACK}$  goes high (T79) at the falling edge of T1. Both  $\overline{CS}$  and  $\overline{SAS}$  could cause  $\overline{DSACK0,1}$  to deassert but only  $\overline{SAS}$  could cause  $\overline{DSACK0,1}$  to become TRI-STATE.

**Note 6:** The timing value includes an RC delay inherent in the test measurement. These signals typically TRI-STATE 7 ns earlier, enabling other devices to drive these lines without contention.

## 9.0 AC and DC Specifications (Continued)

### REGISTER WRITE, BMODE = 1 (Note 1)



TL/F/11719-91

Number	Parameter	20 MHz		25 MHz		Units
		Min	Max	Min	Max	
T56	$\overline{CS}$ Asynchronous Setup to BSCCK (Notes 3, 4)	8		7		ns
T60	$\overline{CS}$ Valid to $\overline{SMACK}$ Low (Notes 2, 3, 4)	1	5	1	5	bcyc
T63	Register Address Setup to $\overline{SAS}$	6		5		ns
T64	Register Address Hold from $\overline{SAS}$	8		7		ns
T69	$\overline{SAS}$ Asynchronous Setup to BSCCK (Notes 3, 4)	7		6		ns
T69a	$\overline{SAS}$ Asynchronous Setup to BSCCK (Notes 3, 5)	5		4		ns
T70a	$\overline{SRW}$ (Write) Setup to $\overline{SAS}$	4		3		ns
T71a	$\overline{SRW}$ (Write) Hold from $\overline{SAS}$	8		7		ns
T75b	BSCCK to $\overline{DSACK0,1}$ Low		22		20	ns
T77	$\overline{CS}$ to $\overline{DSACK0,1}$ High (Note 5)		20		18	ns
T77a	$\overline{SAS}$ to $\overline{DSACK0,1}$ High (Note 5)		31		29	ns
T77b	BSCCK to $\overline{DSACK0,1}$ TRI-STATE (Note 5)		19		17	ns
T78	Skew between $\overline{DSACK0,1}$		3		3	ns
T79a	BSCCK to $\overline{SMACK}$ High (Note 5)		19		17	ns
T81	BSCCK to $\overline{SMACK}$ Low		22		20	ns
T83	Register Write Data Setup to BSCCK	8		7		ns
T84	Register Write Data Hold from BSCCK	14		12		ns
T85a	Minimum $\overline{CS}$ Deassert Time (Notes 2, 3)	1		1		bcyc

**Note 1:** This figure shows a slave access to the SONIC-T when the SONIC-T is idle, or rather not in master mode. If the SONIC-T is a bus master, there will be some differences as noted in the Memory Arbitration/Slave Access diagram. The BSCCK states (T1, T2, etc.) are the equivalent processor states during a slave access.

**Note 2:** bcyc = bus clock cycle time (T3).

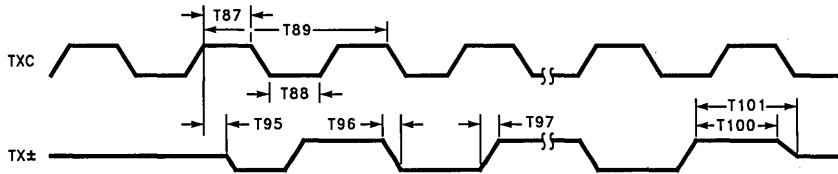
## 9.0 AC and DC Specifications (Continued)

**Note 3:** It is not necessary to meet the setup time for  $\overline{CS}$  and  $\overline{SAS}$  (T56 and T69) since these signals are asynchronously sampled. Meeting the setup time for these signals, however, makes it possible to use T60 to determine when  $\overline{SMACK}$  will be asserted.  $\overline{SAS}$  may be asserted anytime before the next falling edge of the clock that the  $\overline{CS}$  is sampled on (as shown by specification T69). For multiple register accesses,  $\overline{CS}$  can be held low and  $\overline{SAS}$  can be used to delimit the slave cycle (T69a must be met in order to terminate and start another cycle). In this case,  $\overline{SMACK}$  will be asserted as soon as T69 timing is met.

**Note 4:** T60 could range from 1 bus clock minimum to 5 bus clock maximum depending on what state machine the SONIC-T is in when the  $\overline{CS}$  signal is asserted. This timing is not tested, but is guaranteed by design. This specification assumes that both T56 is met for  $\overline{CS}$  and T69 is met for  $\overline{SAS}$ . T60 specification also assumes that there were no wait states in the current master mode access (if  $\overline{CS}$  is asserted when SONIC-T is in Master Mode). If there were wait states, then it would increase the T60 further.

**Note 5:** It is not necessary to meet the setup time for  $\overline{SAS}$  (T69a) since this signal is asynchronously sampled. Meeting the setup time for this signal, however, will ensure  $\overline{DSACK0,1}$  becomes TRI-STATE (77b) and  $\overline{SMACK}$  goes high (T79) at the falling edge of T1. Both  $\overline{CS}$  and  $\overline{SAS}$  could cause  $\overline{DSACK0,1}$  to deassert but only  $\overline{SAS}$  could cause  $\overline{DSACK0,1}$  to become TRI-STATE.

### ENDEC TRANSMIT TIMING



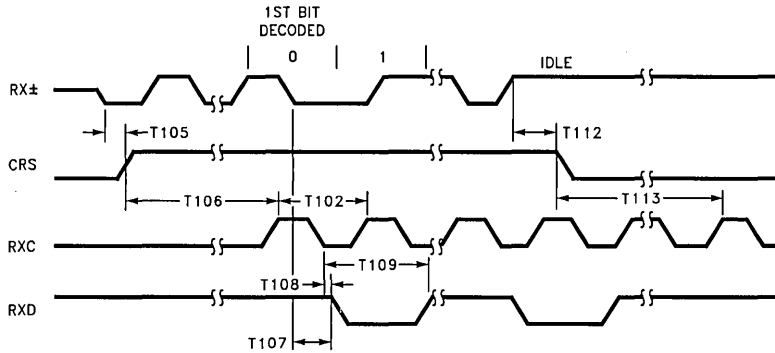
TL/F/11719-76

Number	Parameter	Min	Max	Units
T87	Transmit Clock High Time (Note 1)	40		ns
T88	Transmit Clock Low Time (Note 1)	40		ns
T89	Transmit Clock Cycle Time (Note 1)	99.99	100.01	ns
T95	Transmit Output Delay (Note 1)		55	ns
T96	Transmit Output Fall Time (80% to 20%, Note 1)		7	ns
T97	Transmit Output Rise Time (20% to 80%, Note 1)		7	ns
T98	Transmit Output Jitter (Not Shown)	0.5 Typ		ns
T100	Transmit Output High before Idle (Half Step)	200		ns
T101	Transmit Output Idle Time (Half Step)		8000	ns

**Note 1:** This specification is provided for information only and is not tested.

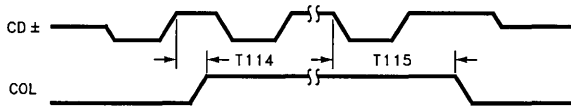
## 9.0 AC and DC Specifications (Continued)

### ENDEC RECEIVE TIMING (INTERNAL ENDEC MODE)



TL/F/11719-77

### ENDEC COLLISION TIMING



TL/F/11719-78

Number	Parameter	Min	Max	Units
T102	Receive Clock Duty Cycle Time (Note 1)	40	60	ns
T105	Carrier Sense on Time		70	ns
T106	Data Acquisition Time		700	ns
T107	Receive Data Output Delay		150	ns
T108	Receive Data Valid from RXC		10	ns
T109	Receive Data Stable Valid Time	90		ns
T112	Carrier Sense Off Delay		250	ns
T113	Minimum Number of RXCs after CRS Low (Note 3)	5		rcyc
T114	Collision Turn On Time		55	ns
T115	Collision Turn Off Time		250	ns

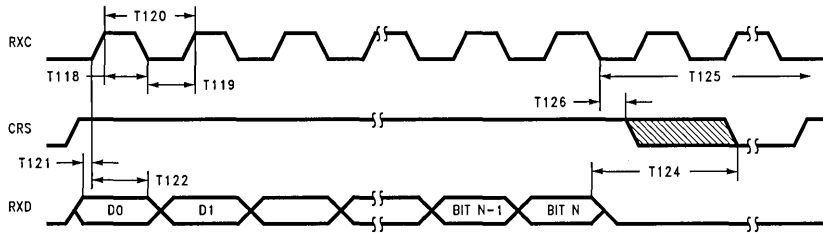
**Note 1:** This parameter is measured at the 50% point of each clock edge.

**Note 2:** When CRSi goes low, it remains low for a minimum of 2 receive clocks (RXC).

**Note 3:** rcyc = receive clocks.

## 9.0 AC and DC Specifications (Continued)

### ENDEC-MAC SERIAL TIMING FOR RECEPTION (EXTERNAL ENDEC MODE)



TL/F/11719-92

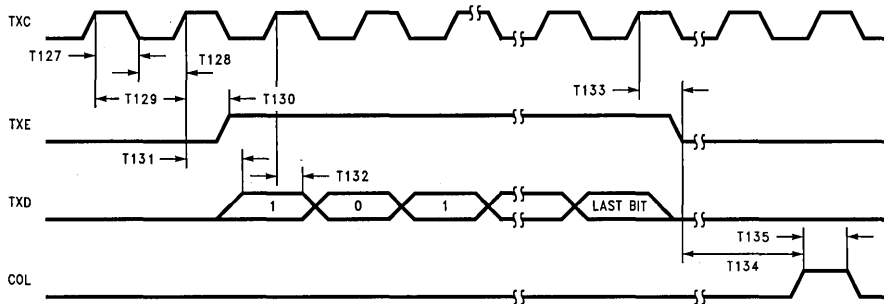
Number	Parameter	Min	Max	Units
T118	Receive Clock High Time	40		ns
T119	Receive Clock Low Time	40		ns
T120	Receive Clock Cycle Time	90	110	ns
T121	RXD Setup to RXC	20		ns
T122	RXD Hold from RXC	15		ns
T124	Maximum Allowed Dribble Bits		6	Bits
T125	Receive Recovery Time (Note 2)			
T126	RXC to Carrier Sense Low (Notes 1, 3)		1	rcyc

**Note 1:** tcyc = transmit clocks, rcyc = receive clocks, bcyc = T3.

**Note 2:** This parameter refers to longest time (not including wait-states) the SONIC™ requires to perform its end of receive processing and be ready for the next start of frame delimiter. This time is 4 + 36 ccyc bcyc. This is guaranteed by design and is not tested.

**Note 3:** To ensure proper receive operation, a minimum of 5 RXCs after CRS low are required.

### ENDEC-MAC SERIAL TIMING FOR TRANSMIT (NO COLLISION)



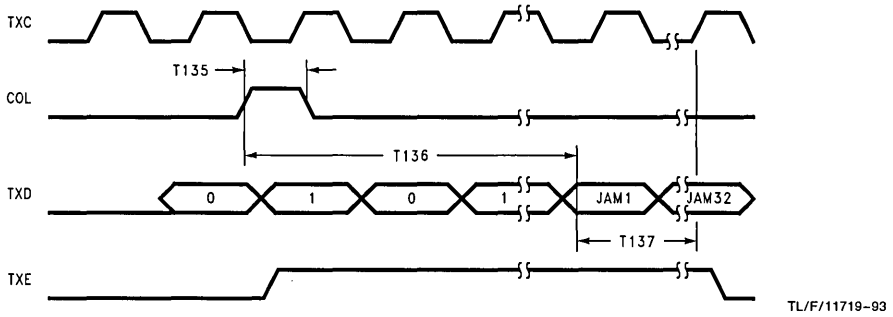
TL/F/11719-86

Number	Parameter	Min	Max	Units
T127	Transmit Clock High Time	40		ns
T128	Transmit Clock Low Time	40		ns
T129	Transmit Clock Cycle Time	90	110	ns
T130	TXC to TXE High		40	ns
T131	TXC to TXD Valid		40	ns
T132	TXD Hold Time from TXC	0		ns
T133	TXC to TXE Low		40	ns
T134	TXE Low to Start of CD Heartbeat (Note 1)		64	tcyc
T135	Collision Detect Width (Note 1)	2		tcyc

**Note 1:** tcyc = transmit clock.

**9.0 AC and DC Specifications** (Continued)

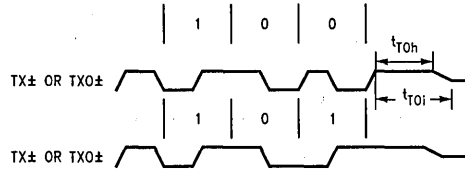
**ENDEC-MAC SERIAL TIMING FOR TRANSMISSION (COLLISION)**



Number	Parameter	Min	Max	Units
T135	Collision Detect Width (Note 1)	2		tcyc
T136	Delay from Collision		8	tcyc
T137	JAM Period		32	tcyc

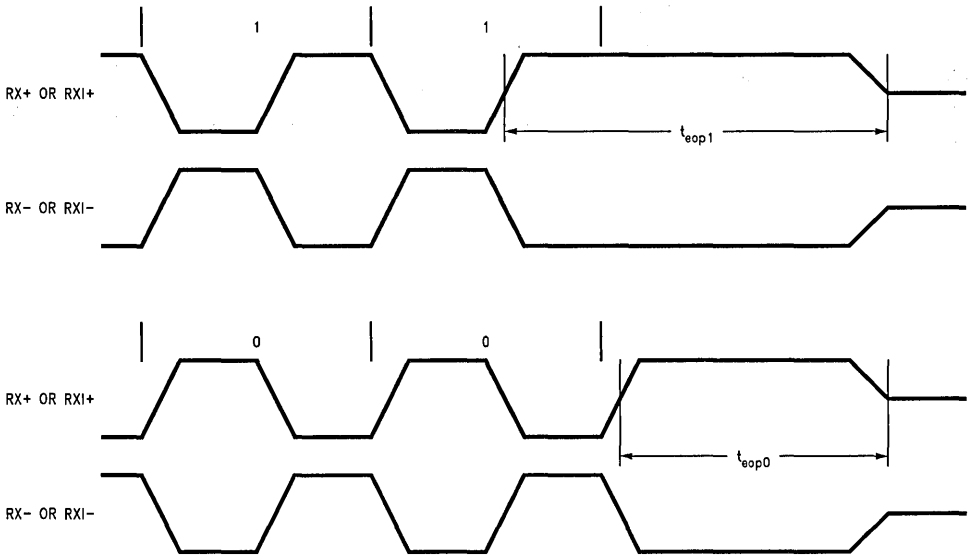
**Note 1:** tcyc = transmit clock.

### 9.0 AC and DC Specifications (Continued)



TL/F/11719-94

Symbol	Parameter	Min	Max	Units
$t_{TOH}$	Transmit Output High before Idle	200		ns
$t_{TOI}$	Transmit Output Idle Time	8000		ns



TL/F/11719-95

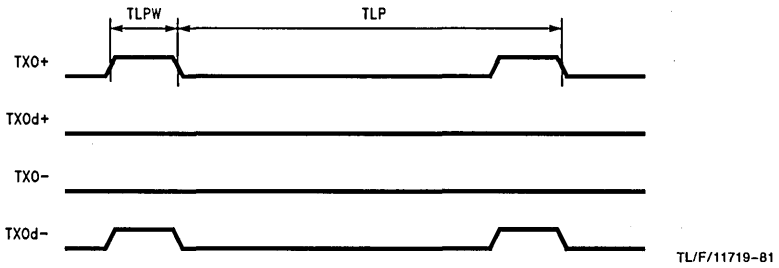
Symbol	Parameter	Min	Max	Units
$t_{eop1}$	Transmit End of Packet Hold Time after Logic "1" (Note 1)	225		ns
$t_{eop0}$	Transmit End of Packet Hold Time after Logic "0" (Note 1)	225		ns

**Note 1:** This parameter is guaranteed by design and is not tested.



## 9.0 AC and DC Specifications (Continued)

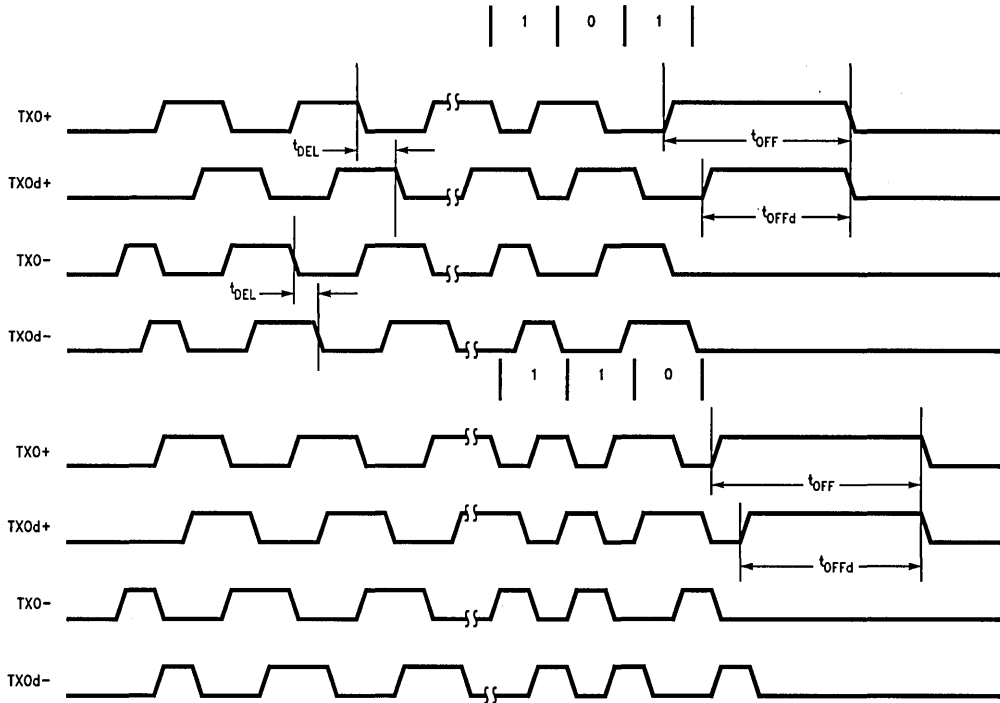
### LINK PULSE TIMING



TL/F/11719-81

Symbol	Parameter	Min	Max	Units
$t_{lp}$	Time between Link Output Pulses	8	24	ms
$t_{lpw}$	Link Integrity Output Pulse Width	80	130	ns

### TPI TRANSMIT TIMING (End of Packet)



TL/F/11719-82

Symbol	Parameter	Min	Max	Units
$t_{del}$	Pre-Emphasis Output Delay (TX0± to TX0±) (Note 1)	46	54	ns
$t_{off}$	Transmit Hold Time at End of Packet (TX0±) (Note 1)	250		ns
$t_{offd}$	Transmit Hold Time at End of Packet (TX0d±) (Note 1)	200		ns

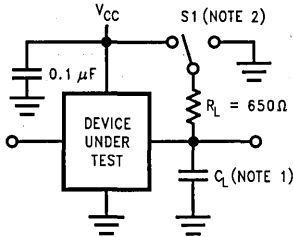
**Note 1:** This parameter is guaranteed by design and is not tested.

## 10.0 AC Timing Test Conditions

All specifications are valid only if the mandatory isolation is employed and all differential signals are taken to be at the AUI side of the pulse transformer.

Input Pulse Levels (TTL/CMOS)	GND to 3.0V
Input Rise and Fall Times (TTL/CMOS)	5 ns
Input and Output Reference Levels (TTL/CMOS)	1.5V
Input Pulse Levels (Diff.)	-350 mV to -1315 mV
Input and Output Reference Levels (Diff.)	50% Point of the Differential
TRI-STATE Reference Levels	Float ( $\Delta V$ ) $\pm 0.5V$

### OUTPUT LOAD (See Figure below)



TL/F/11719-83

**Note 1:** 50 pF, includes scope and jig capacitance.

**Note 2:** S1 = Open for timing test for push pull outputs.

S1 =  $V_{CC}$  for  $V_{OL}$  test.

S1 = GND for  $V_{OH}$  test.

S1 =  $V_{CC}$  for High Impedance to active low and active low to High Impedance measurements.

S1 = GND for High Impedance to active high and active High to High Impedance measurements.

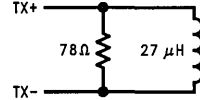
## Pin Capacitance $T_A = 25^\circ C, f = 1 \text{ MHz}$

Symbol	Parameter	Typ	Units
$C_{IN}$	Input Capacitance	7	pF
$C_{OUT}$	Output Capacitance	7	pF

### DERATING FACTOR

Output timing is measured with a purely capacitive load of 50 pF. The following correction factor can be used for other loads:  $C_L \geq 50 \text{ pF} = 0.05 \text{ ns/pF}$ .

### AUI Transmit Test Load



TL/F/11719-84

**Note:** In the above diagram, the TX+ and TX- signals are taken from the AUI side of the isolation (pulse transformer). The pulse transformer used for all testing is a 100  $\mu H \pm 0.1\%$  Pulse Engineering PE64103.

# DP83932C-20/25/33 MHz SONIC™ Systems-Oriented Network Interface Controller

## General Description

The SONIC (Systems-Oriented Network Interface Controller) is a second-generation Ethernet Controller designed to meet the demands of today's high-speed 32- and 16-bit systems. Its system interface operates with a high speed DMA that typically consumes less than 3% of the bus bandwidth (25 MHz bus clock). Selectable bus modes provide both big and little endian byte ordering and a clean interface to standard microprocessors. The linked-list buffer management system of SONIC offers maximum flexibility in a variety of environments from PC-oriented adapters to high-speed motherboard designs. Furthermore, the SONIC integrates a fully-compatible IEEE 802.3 Encoder/Decoder (ENDEC) allowing for a simple 2-chip solution for Ethernet when the SONIC is paired with the DP8392 Coaxial Transceiver Interface or a 10BASE-T transceiver.

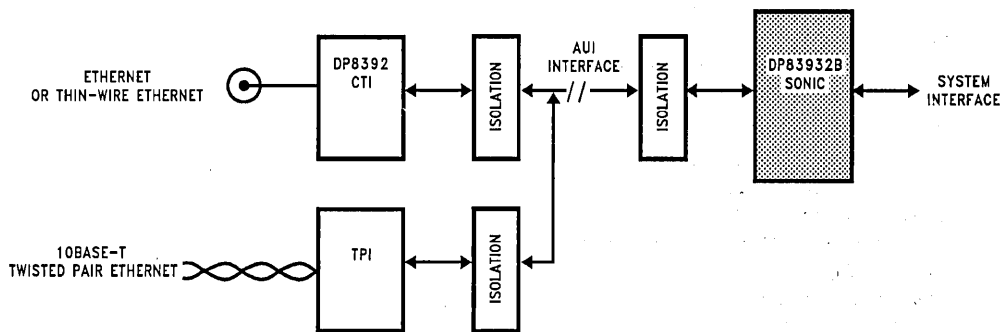
For increased performance, the SONIC implements a unique buffer management scheme to efficiently process, receive and transmit packets in system memory. No intermediate packet copy is necessary. The receive buffer management uses three areas in memory for (1) allocating additional resources, (2) indicating status information, and (3) buffering packet data. During reception, the SONIC stores packets in the buffer area, then indicates receive status and control information in the descriptor area. The system allocates more memory resources to the SONIC by adding descriptors to the memory resource area. The transmit buffer

management uses two areas in memory: one for indicating status and control information and the other for fetching packet data. The system can create a transmit queue allowing multiple packets to be transmitted from a single transmit command. The packet data can reside on any arbitrary byte boundary and can exist in several non-contiguous locations.

## Features

- 32-bit non-multiplexed address and data bus
- High-speed, interruptible DMA
- Linked-list buffer management maximizes flexibility
- Two independent 32-byte transmit and receive FIFOs
- Bus compatibility for all standard microprocessors
- Supports big and little endian formats
- Integrated IEEE 802.3 ENDEC
- Complete address filtering for up to 16 physical and/or multicast addresses
- 32-bit general-purpose timer
- Full-duplex loopback diagnostics
- Fabricated in low-power CMOS
- 132 PQFP package
- Full network management facilities support the 802.3 layer management standard
- Integrated support for bridge and repeater applications

## System Diagram



TL/F/10492-2

## Table of Contents

### 1.0 FUNCTIONAL DESCRIPTION

- 1.1 IEEE 802.3 ENDEC Unit
  - 1.1.1 ENDEC Operation
  - 1.1.2 Selecting an External ENDEC
- 1.2 MAC Unit
  - 1.2.1 MAC Receive Section
  - 1.2.2 MAC Transmit Section
- 1.3 Data Width and Byte Ordering
- 1.4 FIFO and Control Logic
  - 1.4.1 Receive FIFO
  - 1.4.2 Transmit FIFO
- 1.5 Status and Configuration Registers
- 1.6 Bus Interface
- 1.7 Loopback and Diagnostics
  - 1.7.1 Loopback Procedure
- 1.8 Network Management Functions

### 2.0 TRANSMIT/RECEIVE IEEE 802.3 FRAME FORMAT

- 2.1 Preamble and Start Of Frame Delimiter (SFD)
- 2.2 Destination Address
- 2.3 Source Address
- 2.4 Length/Type Field
- 2.5 Data Field
- 2.6 FCS Field
- 2.7 MAC (Media Access Control) Conformance

### 3.0 BUFFER MANAGEMENT

- 3.1 Buffer Management Overview
- 3.2 Descriptor Areas
  - 3.2.1 Naming Convention for Descriptors
  - 3.2.2 Abbreviations
  - 3.2.3 Buffer Management Base Addresses
- 3.3 Descriptor Data Alignment
- 3.4 Receive Buffer Management
  - 3.4.1 Receive Resource Area (RRA)
  - 3.4.2 Receive Buffer Area (RBA)
  - 3.4.3 Receive Descriptor Area (RDA)
  - 3.4.4 Receive Buffer Management Initialization
  - 3.4.5 Beginning of Reception
  - 3.4.6 End of Packet Processing
  - 3.4.7 Overflow Conditions
- 3.5 Transmit Buffer Management
  - 3.5.1 Transmit Descriptor Area (TDA)
  - 3.5.2 Transmit Buffer Area (TBA)
  - 3.5.3 Preparing to Transmit
  - 3.5.4 Dynamically Adding TDA Descriptors

### 4.0 SONIC REGISTERS

- 4.1 The CAM Unit
  - 4.1.1 The Load CAM Command
- 4.2 Status/Control Registers
- 4.3 Register Description
  - 4.3.1 Command Register
  - 4.3.2 Data Configuration Register
  - 4.3.3 Receive Control Register
  - 4.3.4 Transmit Control Register
  - 4.3.5 Interrupt Mask Register
  - 4.3.6 Interrupt Status Register
  - 4.3.7 Data Configuration Register 2
  - 4.3.8 Transmit Registers
  - 4.3.9 Receive Registers
  - 4.3.10 CAM Registers
  - 4.3.11 Tally Counters
  - 4.3.12 General Purpose Timer
  - 4.3.13 Silicon Revision Register

### 5.0 BUS INTERFACE

- 5.1 Pin Configurations
- 5.2 Pin Description
- 5.3 System Configuration
- 5.4 Bus Operations
  - 5.4.1 Acquiring the Bus
  - 5.4.2 Block Transfers
  - 5.4.3 Bus Status
  - 5.4.4 Bus Mode Compatibility
  - 5.4.5 Master Mode Bus Cycles
  - 5.4.6 Bus Exceptions (Bus Retry)
  - 5.4.7 Slave Mode Bus Cycle
  - 5.4.8 On-Chip Memory Arbiter
  - 5.4.9 Chip Reset

### 6.0 NETWORK INTERFACING

- 6.1 Manchester Encoder and Differential Driver
  - 6.1.1 Manchester Decoder
  - 6.1.2 Collision Translator
  - 6.1.3 Oscillator Inputs
  - 6.1.4 Power Supply Considerations

### 7.0 AC AND DC SPECIFICATIONS

### 8.0 AC TIMING TEST CONDITIONS

## 1.0 Functional Description

The SONIC (*Figure 1-1*) consists of an encoder/decoder (ENDEC) unit, media access control (MAC) unit, separate receive and transmit FIFOs, a system buffer management engine, and a user programmable system bus interface unit on a single chip. SONIC is highly pipelined providing maximum system level performance. This section provides a functional overview of SONIC.

### 1.1 IEEE 802.3 ENDEC UNIT

The ENDEC (Encoder/Decoder) unit is the interface between the Ethernet transceiver and the MAC unit. It provides the Manchester data encoding and decoding functions for IEEE 802.3 Ethernet/Thin-Ethernet type local area networks. The ENDEC operations of SONIC are identical to the DP83910A CMOS Serial Network Interface device. During transmission, the ENDEC unit combines non-return-zero (NRZ) data from the MAC section and clock pulses into Manchester data and sends the converted data differentially to the transceiver. Conversely, during reception, an analog PLL decodes the Manchester data to NRZ format and receive clock. The ENDEC unit is a functionally complete Manchester encoder/decoder incorporating a balanced driver and receiver, on-board crystal oscillator, collision signal translator, and a diagnostic loopback. The features include:

- Compatible with Ethernet I and II, IEEE 802.3 10base5 and 10base2
- 10Mb/s Manchester encoding/decoding with receive clock recovery
- Requires no precision components
- Loopback capability for diagnostics
- Externally selectable half or full step modes of operation at transmit output
- Squelch circuitry at the receive and collision inputs reject noise
- Connects to the transceiver (AUI) cable via external pulse transformer

### 1.1.1 ENDEC Operation

The primary function of the ENDEC unit (*Figure 1-2*) is to perform the encoding and decoding necessary for compatibility between the differential pair Manchester encoded data of the transceiver and the Non-Return-to-Zero (NRZ) serial data of the MAC unit data line. In addition to encoding and decoding the data stream, the ENDEC also supplies all the necessary special signals (e.g., collision detect, carrier sense, and clocks) to the MAC unit. The signals provided to the MAC unit from the on-chip ENDEC are also provided as outputs to the user.

#### Manchester Encoder and Differential Output Driver:

During transmission to the network, the ENDEC unit translates the NRZ serial data from the MAC unit into differential pair Manchester encoded data on the Coaxial Transceiver Interface (e.g., National's DP8392) transmit pair. To perform this operation the NRZ bit stream from the MAC unit is passed through the Manchester encoder block of the ENDEC unit. Once the bit stream is encoded, it is transmitted out differentially to the transmit differential pair through the transmit driver.

**Manchester Decoder:** During reception from the network, the differential receive data from the transceiver (e.g., the DP8392) is converted from Manchester encoded data into NRZ serial data and a receive clock, which are sent to the receive data and clock inputs of the MAC unit. To perform this operation the signal, once received by the differential receiver, is passed to the phase locked loop (PLL) decoder block. The PLL decodes the data and generates a data receive clock and a NRZ serial data stream to the MAC unit.

**Special Signals:** In addition to performing the Manchester encoding and decoding function, the ENDEC unit provides control and clocking signals to the MAC unit. The ENDEC sends a carrier sense (CRS) signal that indicates to the MAC unit that data is present from the network on the ENDEC's receive differential pair. The MAC unit is also provided with a collision detection signal (COL) that informs the MAC unit that a collision is taking place somewhere on

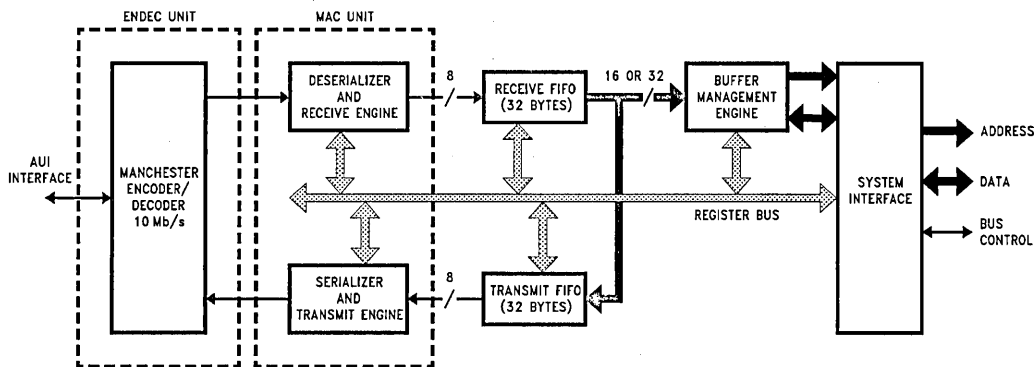


FIGURE 1-1. SONIC Block Diagram

TL/F/10492-1

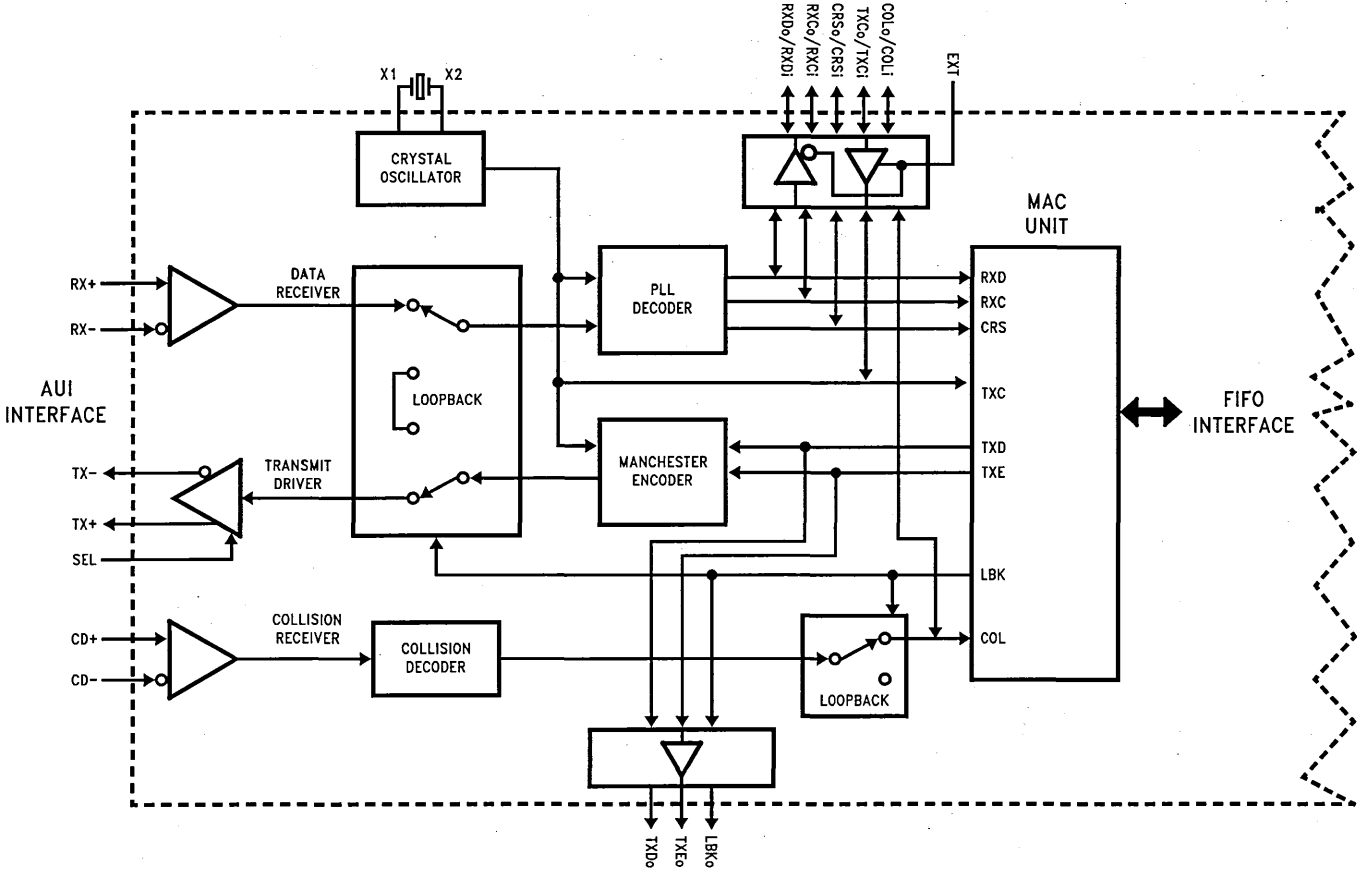


FIGURE 1-2. Block Diagram of Ethernet ENDEC

1-640

## 1.0 Functional Description (Continued)

the network. The ENDEC section detects this when its collision receiver detects a 10 MHz signal on the differential collision input pair. The ENDEC also provides both the receive and transmit clocks to the MAC unit. The transmit clock is one half of the oscillator input. The receive clock is extracted from the input data by the PLL.

**Oscillator:** The oscillator generates the 10 MHz transmit clock signal for network timing. The oscillator is controlled by a parallel resonant crystal or by an external clock (see Section 6.1.3). The 20 MHz output of the oscillator is divided by 2 to generate the 10 MHz transmit clock (TXC) for the MAC section. The oscillator provides an internal clock signal for the encoding and decoding circuits.

**Loopback Functions:** The SONIC provides three loopback modes. These modes allow loopback testing at the MAC, ENDEC and external transceiver level (see Section 1.7 for details). It is important to note that when the SONIC is transmitting, the transmitted packet will always be looped back by the external transceiver. The SONIC takes advantage of this to monitor the transmitted packet. See the explanation of the Receive State Machine in Section 1.2.1 for more information about monitoring transmitted packets.

### 1.1.2 Selecting An External ENDEC

An option is provided on SONIC to disable the on-chip ENDEC unit and use an external ENDEC. The internal IEEE 802.3 ENDEC can be bypassed by connecting the EXT pin to V<sub>CC</sub> (EXT = 1). In this mode the MAC signals are redirected, allowing an external ENDEC to be used. See Section 5.2 for the alternate pin definitions.

## 1.2 MAC UNIT

The MAC (Media Access Control) unit performs the media access control functions for transmitting and receiving packets over Ethernet. During transmission, the MAC unit frames information from the transmit FIFO and supplies serialized data to the ENDEC unit. During reception, the incoming information from the ENDEC unit is deserialized, the frame checked for valid reception, and the data is transferred to the receive FIFO. Control and status registers on the SONIC govern the operation of the MAC unit.

### 1.2.1 MAC Receive Section

The receive section (*Figure 1-3*) controls the MAC receive operations during reception, loopback, and transmission. During reception, the deserializer goes active after detecting the one byte SFD (Start of Frame Delimiter) pattern (Section 2.1) consisting of a "10101011" sequence. It then frames the incoming bits into octet boundaries and transfers the

data to the 32-byte receive FIFO. Concurrently the address comparator compares the Destination Address Field to the addresses stored in the chip's CAM address registers (Content Addressable Memory cells). If a match occurs, the deserializer passes the remainder of the packet to the receive FIFO. The packet is decapsulated when the carrier sense input pin (CRS) goes inactive. At the end of reception the receive section checks the following:

- Frame alignment errors
- CRC errors
- Length errors (runt packets)

The appropriate status is indicated in the Receive Control register (Section 4.3.3). In loopback operations, the receive section operates the same as during normal reception.

During transmission, the receive section remains active to allow monitoring of the self-received packet. The CRC checker operates as normal, and the Source Address field is compared with the CAM address entries. Status of the CRC check and the source address comparison is indicated by the PMB bit in the Transmit Control register (Section 4.3.4). No data is written to the receive FIFO during transmit operations.

The receive section consists of the following blocks detailed below.

**Receive State Machine (RSM):** The RSM insures the proper sequencing for normal reception and self-reception during transmission. When the network is inactive, the RSM remains in an idle state continually monitoring for network activity. If the network becomes active, the RSM allows the deserializer to write data into the receive FIFO. During this state, the following conditions may prevent the complete reception of the packet.

- FIFO Overrun—The receive FIFO has been completely filled before the SONIC could buffer the data to memory.
- CAM Address Mismatch—The packet is rejected because of a mismatch between the destination address of the packet and the address in the CAM.
- Memory Resource Error—There are no more resources (buffers) available for buffering the incoming packets.
- Collision or Other Error—A collision occurred on the network or some other error, such as a CRC error, occurred (this is true if the SONIC has been told to reject packets on a collision, or reject packets with errors).

If these conditions do not occur, the RSM processes the packet indicating the appropriate status in the Receive Control register.

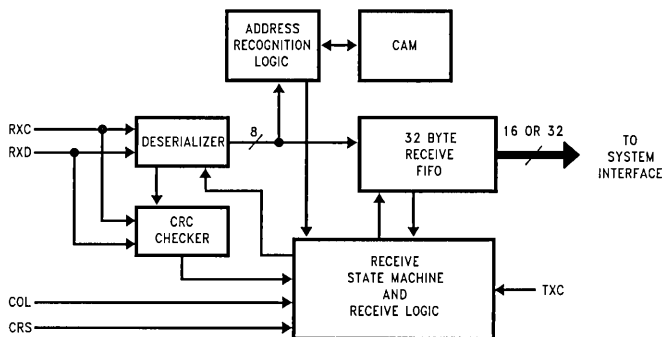


FIGURE 1-3. MAC Receiver

TL/F/10492-4

## 1.0 Functional Description (Continued)

During transmission of a packet from the SONIC, the external transceiver will always loop the packet back to the SONIC. The SONIC will use this to monitor the packet as it is being transmitted. The CRC and source address of the looped back packet are checked with the CRC and source address that were transmitted. If they do not match, an error bit is set in the status of the transmitted packet (see Packet Monitored Bad, PMB, in the Transmit Control Register, Section 4.3.4). Data is not written to the receive FIFO during this monitoring process unless Transceiver Loopback mode has been selected (see Section 1.7).

**Receive Logic:** The receive logic contains the command, control, and status registers that govern the operations of the receive section. It generates the control signals for writing data to the receive FIFO, processes error signals obtained from the CRC checker and the deserializer, activates the "packet reject" signal to the RSM for rejecting packets, and posts the applicable status in the Receive Control register.

**Deserializer:** This section deserializes the serial input data stream and furnishes a byte clock for the address comparator and receive logic. It also synchronizes the CRC checker to begin operation (after SFD is detected), and checks for proper frame alignment with respect to CRS going inactive at the end of reception.

**Address Comparator:** The address comparator latches the Destination Address (during reception or loopback) or Source Address (during transmission) and determines whether the address matches one of the entries in the CAM (Content Addressable Memory).

**CRC Checker:** The CRC checker calculates the 4-byte Frame Check Sequence (FCS) field from the incoming data stream and compares it with the last 4-bytes of the received packet. The CRC checker is active for both normal reception and self-reception during transmission.

**Content Addressable Memory (CAM):** The CAM contains 16 user programmable entries and 1 pre-programmed Broadcast address entry for complete filtering of received packets. The CAM can be loaded with any combination of Physical and Multicast Addresses (Section 2.2). See Section 4.1 for the procedure on loading the CAM registers.

### 1.2.2 MAC Transmit Section

The transmit section (Figure 1-4) is responsible for reading data from the transmit FIFO and transmitting a serial data

stream onto the network in conformance with the IEEE 802.3 CSMA/CD standard. The Transmit Section consists of the following blocks.

**Transmit State Machine (TSM):** The TSM controls the functions of the serializer, preamble generator, and JAM generator. It determines the proper sequence of events that the transmitter follows under various network conditions. If no collision occurs, the transmitter prefixes a 7 byte preamble and 1 byte Start of Frame Delimiter (SFD) consisting of a "10101011" sequence at the beginning of each packet, then sends the serialized data. At the end of the packet, an optional 4-byte CRC pattern is appended. If a collision occurs, the transmitter switches from transmitting data to sending a 4-byte Jam pattern to notify all nodes that a collision has occurred. Should the collision occur during the preamble, the transmitter waits for it to complete before jamming. After the transmission has completed, the transmitter writes status in the Transmit Control register (Section 4.3.4).

**Protocol State Machine:** The protocol state machine assures that the SONIC obeys the CSMA/CD protocol. Before transmitting, this state machine monitors the carrier sense and collision signals for network activity. If another node(s) is currently transmitting, the SONIC defers until the network is quiet, then transmits after its Interframe Gap Timer (9.6  $\mu$ s) has expired. The Interframe Gap time is divided into two portions. During the first 6.4  $\mu$ s, network activity restarts the Interframe Gap timer. Beyond this time, however, network activity is ignored and the state machine waits the remaining 3.2  $\mu$ s before transmitting. If the SONIC experiences a collision during a transmission, the SONIC switches from transmitting data to a 4-byte JAM pattern (4 bytes of all 1's), before ceasing to transmit. The SONIC then waits a random number of slot times (51.2  $\mu$ s) determined by the *Truncated Binary Exponential Backoff Algorithm* before reattempting another transmission. In this algorithm, the number of slot times to delay before the nth retransmission is chosen to be a random integer  $r$  in the range of:

$$0 \leq r \leq 2^k$$

$$\text{where } k = \min(n, 10)$$

If a collision occurs on the 16th transmit attempt, the SONIC aborts transmitting the packet and reports an "Excessive Collisions" error in the Transmit Control register.

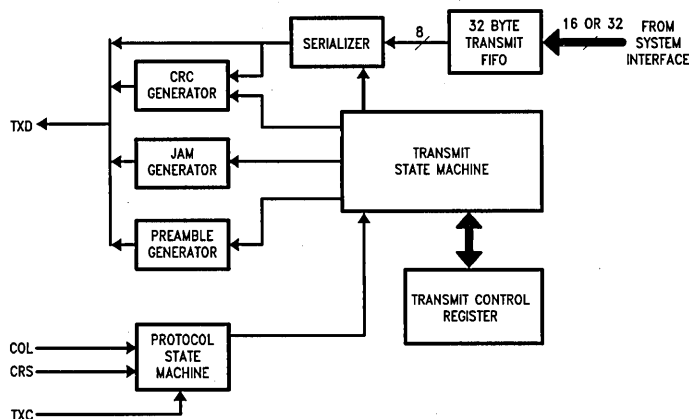


FIGURE 1-4. MAC Transmitter



## 1.0 Functional Description (Continued)

**Serializer:** After data has been written into the 32-byte transmit FIFO, the serializer reads byte wide data from the FIFO and sends a NRZ data stream to the Manchester encoder. The rate at which data is transmitted is determined by the transmit clock (TXC). The serialized data is transmitted after the SFD.

**Preamble Generator:** The preamble generator prefixes a 7 byte alternating "1,0" pattern and a 1 byte "10101011" SFD pattern at the beginning of each packet. This allows receiving nodes to synchronize to the incoming data. The preamble is always transmitted in its entirety even in the event of a collision. This assures that the minimum collision fragment is 96 bits (64 bits of normal preamble, and 4 bytes, or rather 32 bits, of the JAM pattern).

**CRC Generator:** The CRC generator calculates the 4-byte FCS field from the transmitted serial data stream. If enabled, the 4-byte FCS field is appended to the end of the transmitted packet (Section 2.6).

For bridging or switched ethernet applications the CRC Generator can be inhibited by setting bit 13 in the Transmit Control Register (Section 4.3.4). This feature is used when an ethernet segment has already received a packet with a CRC appended and needs to forward it to another ethernet segment.

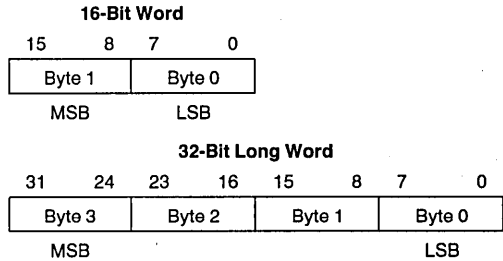
**Jam Generator:** The Jam generator produces a 4-byte pattern of all 1's to assure that all nodes on the network sense the collision. When a collision occurs, the SONIC stops transmitting data and enables the Jam generator. If a collision occurs during the preamble, the SONIC finishes transmitting the preamble before enabling the Jam generator (see Preamble Generator above).

### 1.3 DATA WIDTH AND BYTE ORDERING

The SONIC can be programmed to operate with either 32-bit or 16-bit wide memory. The data width is configured during initialization by programming the DW bit in the Data Configuration Register (DCR, Section 4.3.2). If the 16-bit data path is selected, data is driven on pins D15-D0. The SONIC also provides both Little Endian and Big Endian

byte-ordering capability for compatibility with National/Intel or Motorola microprocessors respectively by selecting the proper level on the BMODE pin. The byte ordering is depicted below.

**Little Endian mode (National/Intel, BMODE = 0):** The byte orientation for received and transmitted data in the Receive Buffer Area (RBA) and Transmit Buffer Area (TBA) of system memory is as follows:



**Big Endian mode (Motorola, BMODE = 1):** The byte orientation for received and transmitted data in the RBA and TBA is as follows:

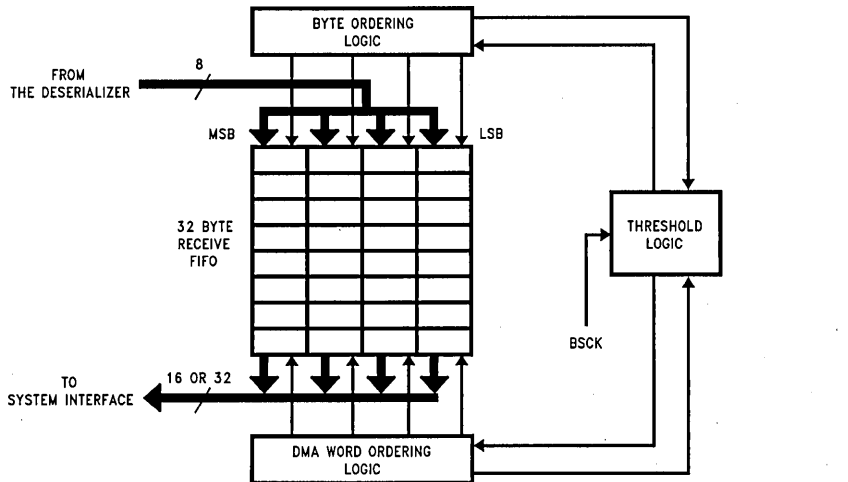
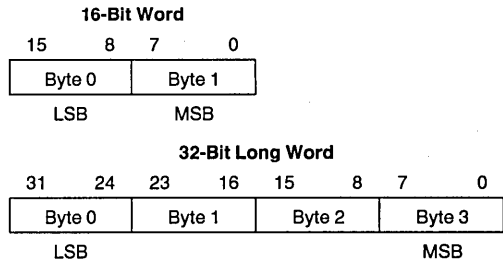


FIGURE 1-5. Receive FIFO

TL/F/10492-6

## 1.0 Functional Description (Continued)

### 1.4 FIFO AND CONTROL LOGIC

The SONIC incorporates two independent 32-byte FIFOs for transferring data to/from the system interface and from/to the network. The FIFOs, providing temporary storage of data, free the host system from the real-time demands on the network.

The way in which the FIFOs are emptied and filled is controlled by the FIFO threshold values and the Block Mode Select bits (BMS, Section 4.3.2). The threshold values determine how full or empty the FIFOs can be before the SONIC will request the bus to get more data from memory or buffer more data to memory. When block mode is set, the number of bytes transferred is set by the threshold value. For example, if the threshold for the receive FIFO is 4 words, then the SONIC will always transfer 4 words from the receive FIFO to memory. If empty/fill mode is set, however, the number of bytes transferred is the number required to fill the transmit FIFO or empty the receive FIFO. More specific information about how the threshold affects reception and transmission of packets is discussed in Sections 1.4.1 and 1.4.2 below.

#### 1.4.1 Receive FIFO

To accommodate the different transfer rates, the receive FIFO (*Figure 1-5*) serves as a buffer between the 8-bit network (deserializer) interface and the 16/32-bit system interface. The FIFO is arranged as a 4-byte wide by 8 deep memory array (8 long words, or 32 bytes) controlled by three sections of logic. During reception, the Byte Ordering logic directs the byte stream from the deserializer into the FIFO using one of four write pointers. Depending on the selected byte-ordering mode, data is written either least significant byte first or most significant byte first to accommodate little or big endian byte-ordering formats respectively.

As data enters the FIFO, the Threshold Logic monitors the number of bytes written in from the deserializer. The programmable threshold (RFT1,0 in the Data Configuration Register) determines the number of words (or long words) written into the FIFO from the MAC unit before a DMA request for system memory occurs. When the threshold is reached, the Threshold Logic enables the Buffer Management Engine to read a programmed number of 16- or 32-bit words (depending upon the selected data width) from the FIFO and transfers them to the system interface (the system memory) using DMA. The threshold is reached when the number of bytes in the receive FIFO is greater than the value of the threshold. For example, if the threshold is 4 words (8 bytes), then the Threshold Logic will not cause the Buffer Management Engine to write to memory until there are more than 8 bytes in the FIFO.

The Buffer Management Engine reads either the upper or lower half (16 bits) of the FIFO in 16-bit mode or reads the complete long word (32 bits) in 32-bit mode. If, after the transfer is complete, the number of bytes in the FIFO is less than the threshold, then the SONIC is done. This is always the case when the SONIC is in empty/fill mode. If, however, for some reason (e.g. latency on the bus) the number of bytes in the FIFO is still greater than the threshold value, the Threshold Logic will cause the Buffer Management Engine to do a DMA request to write to memory again. This later case is usually only possible when the SONIC is in block mode.

When in block mode, each time the SONIC requests the bus, only a number of bytes equal to the threshold value will be transferred. The Threshold Logic continues to monitor

the number of bytes written in from the deserializer and enables the Buffer Management Engine every time the threshold has been reached. This process continues until the end of the packet.

Once the end of the packet has been reached, the serializer will fill out the last word (16-bit mode) or long word (32-bit mode) if the last byte did not end on a word or long word boundary respectively. The fill byte will be 0FFh. Immediately after the last byte (or fill byte) in the FIFO, the received packets status will be written into the FIFO. The entire packet, including any fill bytes and the received packet status will be buffered to memory. When a packet is buffered to memory by the Buffer Management Engine, it is always taken from the FIFO in words or long words and buffered to memory on word (16-bit mode) or long word (32-bit mode) boundaries. Data from a packet cannot be buffered on odd byte boundaries for 16-bit mode, and odd word boundaries for 32-bit mode (see Section 3.3). For more information on the receive packet buffering process, see Section 3.4.

#### 1.4.2 Transmit FIFO

Similar to the Receive FIFO, the Transmit FIFO (*Figure 1-6*) serves as a buffer between the 16/32-bit system interface and the network (serializer) interface. The Transmit FIFO is also arranged as a 4 byte by 8 deep memory array (8 long words or 32 bytes) controlled by three sections of logic. Before transmission can begin, the Buffer Management Engine fetches a programmed number of 16- or 32-bit words from memory and transfers them to the FIFO. The Buffer Management Engine writes either the upper or lower half (16 bits) into the FIFO for 16-bit mode or writes the complete long word (32 bits) during 32-bit mode.

The Threshold logic monitors the number of bytes as they are written into the FIFO. When the threshold has been reached, the Transmit Byte Ordering state machine begins reading bytes from the FIFO to produce a continuous byte stream for the serializer. The threshold is met when the number of bytes in the FIFO is greater than the value of the threshold. For example, if the transmit threshold is 4 words (8 bytes), the Transmit Byte Ordering state machine will not begin reading bytes from the FIFO until there are 9 or more bytes in the buffer. The Buffer Management Engine continues replenishing the FIFO until the end of the packet. It does this by making multiple DMA requests to the system interface. Whenever the number of bytes in the FIFO is equal to or less than the threshold value, the Buffer Management Engine will do a DMA request. If block mode is set, then after each request has been granted by the system, the Buffer Management Engine will transfer a number of bytes equal to the threshold value into the FIFO. If empty/fill mode is set, the FIFO will be completely filled in one DMA request.

Since data may be organized in big or little endian byte ordering format, the Transmit Byte Ordering state machine uses one of four read pointers to locate the proper byte within the 4 byte wide FIFO. It also determines the valid number of bytes in the FIFO. For packets which begin or end at odd bytes in the FIFO, the Buffer Management Engine writes extraneous bytes into the FIFO. The Transmit Byte Ordering state machine detects these bytes and only transfers the valid bytes to the serializer. The Buffer Management Engine can read data from memory on any byte boundary (see Section 3.3). See Section 3.5 for more information on transmit buffering.

## 1.0 Functional Description (Continued)

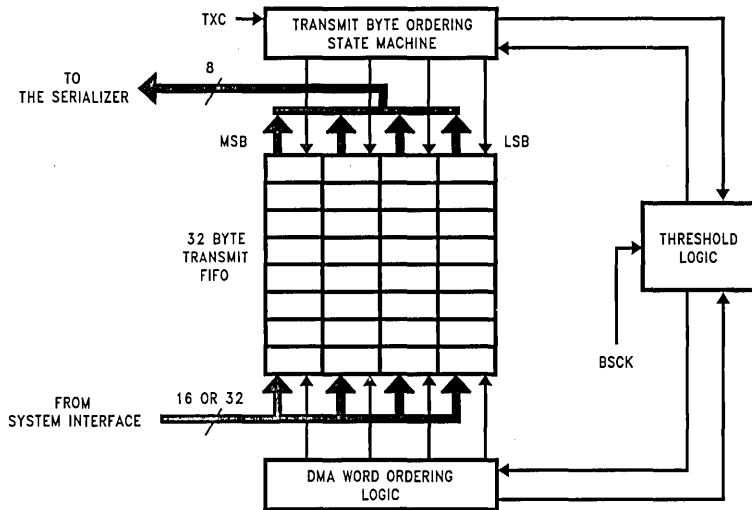


FIGURE 1-6. Transmit FIFO

### 1.5 STATUS AND CONFIGURATION REGISTERS

The SONIC contains a set of status/control registers for conveying status and control information to/from the host system. The SONIC uses these registers for loading commands generated from the system, indicating transmit and receive status, buffering data to/from memory, and providing interrupt control. Each register is 16 bits in length. See Section 4.0 for a description of the registers.

### 1.6 BUS INTERFACE

The system interface (Figure 1-7) consists of the pins necessary for interfacing to a variety of buses. It includes the I/O drivers for the data and address lines, bus access control for standard microprocessors, ready logic for synchronous or asynchronous systems, slave access control, interrupt control, and shared-memory access control. The functional signal groups are shown in Figure 1-7. See Section 5.0 for a complete description of the SONIC bus interface.

### 1.7 LOOPBACK AND DIAGNOSTICS

The SONIC furnishes three loopback modes for self-testing from the controller interface to the transceiver interface. The loopback function is provided to allow self-testing of the chip's internal transmit and receive operations. During loopback, transmitted packets are routed back to the receive section of the SONIC where they are filtered by the address recognition logic and buffered to memory if accepted. Transmit and receive status and interrupts remain active during loopback. This means that when using loopback, it is as if the packet was transmitted and received by two separate chips that are connected to the same bus and memory.

**MAC Loopback:** Transmitted data is looped back at the MAC. Data is not sent from the MAC to either the internal ENDEC or an external ENDEC (the external ENDEC interface pins will not be driven), hence, data is not transmitted from the chip. Even though the ENDEC is not used in MAC loopback, the ENDEC clock (an oscillator or crystal for the

internal ENDEC or TXC for an external ENDEC) must be driven. Network activity, such as a collision, does not affect MAC loopback. CSMA/CD MAC protocol is not completely followed in MAC loopback.

**ENDEC Loopback:** Transmitted data is looped back at the ENDEC. If the internal ENDEC is used, data is switched from the transmit section of the ENDEC to the receive section (Figure 1-2). Data is not transmitted from the chip and the collision lines,  $CD \pm$ , are ignored, hence, network activity does not affect ENDEC loopback. The LBK signal from the MAC tells the internal ENDEC to go into loopback mode. If an external ENDEC is used, it should operate in loopback mode when the LBK signal is asserted. CSMA/CD MAC protocol is followed even though data is not transmitted from the chip.

**Transceiver Loopback:** Transmitted data is looped back at the external transceiver (which is always the case regardless of the SONIC's loopback mode). CSMA/CD MAC protocol is followed since data will be transmitted from the chip. This means that transceiver loopback is affected by network activity. In normal operations, the SONIC only monitors the packet that is looped back by the transceiver, but does not fill the receive FIFO and buffer the packet.

#### 1.7.1 Loopback Procedure

The following procedure describes the loopback operation.

1. Initialize the Transmit and Receive Area as described in Sections 3.4 and 3.5.
2. Load one of the CAM address registers (see Section 4.1), with the Destination Address of the packet if you are verifying the SONIC's address recognition capability.
3. Load one of the CAM address registers with the Source Address of the packet if it is different than the Destination Address to avoid getting a Packet Monitored Bad (PMB) error in the Transmit status (see Section 4.3.4).

### 1.0 Functional Description (Continued)

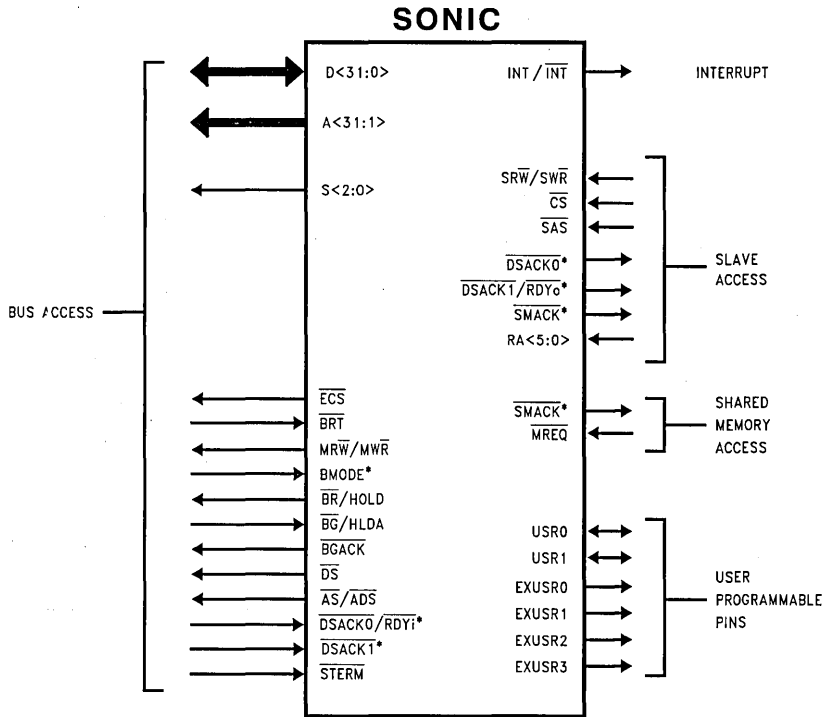
4. Program the Receive Control register with the desired receive filter and the loopback mode (LB1, LB0).
5. Issue the transmit command (TXP) and enable the receiver (RXEN) in the Command register.

The SONIC completes the loopback operation after the packet has been completely received (or rejected if there is an address mismatch). The Transmit Control and Receive Control registers treat the loopback packet as in normal operation and indicate status accordingly. Interrupts are also generated if enabled in the Interrupt Mask register.

**Note:** For MAC Loopback, only one packet may be queued for proper operation. This restriction occurs because the transmit MAC section, which does not generate an Interframe Gap time (IFG) between transmitted packets, does not allow the receive MAC section to update receive status. There are no restrictions for the other loopback modes.

### 1.8 NETWORK MANAGEMENT FUNCTIONS

The SONIC fully supports the Layer Management IEEE 802.3 standard to allow a node to monitor the overall performance of the network. These statistics are available on a per packet basis at the end of reception or transmission. In addition, the SONIC provides three tally counters to tabulate CRC errors, Frame Alignment errors, and missed packets. Table 1-1 shows the statistics indicated by the SONIC.



TL/F/10492-8

\*Note: DSACK0,1 are used for both Bus and Slave Access Control and are bidirectional. SMACK is used for both Slave access and shared memory access. The BMODE pin selects between National/Intel or Motorola type buses.

FIGURE 1-7. SONIC Bus Interface Signals

# 1.0 Functional Description (Continued)

**TABLE 1-1. Network Management Statistics**

Statistic	Register Used	Bits Used
Frames Transmitted OK	TCR (Note)	PTX
Single Collision Frames	(Note)	NC0-NC4
Multiple Collision Frames	(Note)	NC0-NC4
Collision Frames	(Note)	NC0-NC4
Frames with Deferred Transmissions	TCR (Note)	DEF
Late Collisions	TCR (Note)	OWC
Excessive Collisions	TCR (Note)	EXC
Excessive Deferral	TCR (Note)	EXD
Internal MAC Transmit Error	TCR (Note)	BCM, FU
Frames Received OK	RCR (Note)	PRX
Multicast Frames Received OK	RCR (Note)	MC
Broadcast Frames Received OK	RCR (Note)	BC
Frame Check Sequence Errors	CRCT RCR	All CRC
Alignment Errors	FAET RCR	All FAE
Frame Lost Due to Internal MAC Receive Error	MPT ISR	All RFO

**Note:** The number of collisions and the contents of the Transmit Control register are posted in the TXpkt.status field (see Section 3.5.1.2). The contents of the Receive Control register are posted in the RXpkt.status field (see Section 3.4.3).

## 2.0 Transmit/Receive IEEE 802.3 Frame Format

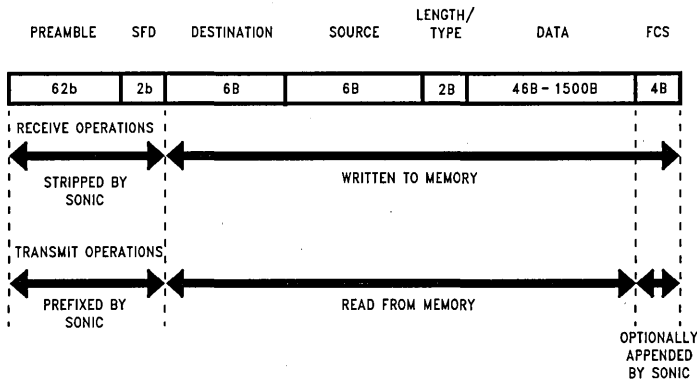
A standard IEEE 802.3 packet (*Figure 2-1*) consists of the following fields: preamble, Start of Frame Delimiter (SFD), destination address, source address, length, data and Frame Check Sequence (FCS). The typical format is shown in *Figure 2-1*. The packets are Manchester encoded and decoded by the ENDEC unit and transferred serially to/from the MAC unit using NRZ data with a clock. All fields are of fixed length except for the data field. The SONIC generates and appends the preamble, SFD and FCS field during transmission. The Preamble and SFD fields are stripped during reception. (The CRC is passed through to buffer memory during reception.)

### 2.1 PREAMBLE AND START OF FRAME DELIMITER (SFD)

The Manchester encoded alternating 1,0 preamble field is used by the ENDEC to acquire bit synchronization with an incoming packet. When transmitted, each packet contains 62 bits of an alternating 1,0 preamble. Some of this preamble may be lost as the packet travels through the network. Byte alignment is performed when the Start of Frame Delimiter (SFD) pattern, consisting of two consecutive 1's, is detected.

### 2.2 DESTINATION ADDRESS

The destination address indicates the destination of the packet on the network and is used to filter unwanted pack-



**Note:** B = bytes  
b = bits

**FIGURE 2-1. IEEE 802.3 Packet Structure**

TL/F/10492-9

## 2.0 Transmit/Receive IEEE 802.3 Frame Format (Continued)

ets from reaching a node. There are three types of address formats supported by the SONIC: Physical, Multicast, and Broadcast.

**Physical Address:** The physical address is a unique address that corresponds only to a single node. All physical addresses have the LSB of the first byte of the address set to "0". These addresses are compared to the internally stored CAM (Content Addressable Memory) address entries. All bits in the destination address must match an entry in the CAM in order for the SONIC to accept the packet.

**Multicast Address:** Multicast addresses, which have the LSB of the first byte of the address set to "1", are treated similarly as Physical addresses, i.e., they must match an entry in the CAM. This allows perfect filtering of Multicast packets and eliminates the need for a hashing algorithm for mapping Multicast packets.

**Broadcast Address:** If the address consists of all 1's, it is a Broadcast address, indicating that the packet is intended for all nodes.

The SONIC also provides a promiscuous mode which allows reception of all physical address packets. Physical, Multicast, Broadcast, and promiscuous address modes can be selected via the Receive Control register.

### 2.3 SOURCE ADDRESS

The source address is the physical address of the sending node. Source addresses cannot be multicast or broadcast addresses. This field must be passed to the SONIC's transmit buffer from the system software. During transmission, the SONIC compares the Source address with its internal CAM address entries before monitoring the CRC of the self-received packet. If the source address of the packet transmitted does not match a value in the CAM, the packet monitored bad flag (PMB) will be set in the transmit status field of the transmit descriptor (see Sections 3.5.1.2 and 4.3.4). The SONIC does not provide Source Address insertion. However, a transmit descriptor fragment, containing only the Source Address, may be created for each packet. (See Section 3.5.1.)

### 2.4 LENGTH/TYPE FIELD

For IEEE 802.3 type packets, this field indicates the number of bytes that are contained in the data field of the packet. For Ethernet I and II networks, this field indicates the type of packet. The SONIC does not operate on this field.

### 2.5 DATA FIELD

The data field has a variable octet length ranging from 46 to 1500 bytes as defined by the Ethernet specification. Messages longer than 1500 bytes need to be broken into multiple packets for IEEE 802.3 networks. Data fields shorter than 46 bytes require appending a pad to bring the complete frame length to 64 bytes. If the data field is padded, the number of valid bytes are indicated in the length field. The SONIC does not append pad bytes for short packets during transmission, nor check for oversize packets during reception. However, the user's driver software can easily append the pad by lengthening the TXpkt.pkt\_size field and TXpkt.frag\_size field(s) to at least 64 bytes (see Section 3.5.1). While the Ethernet specification defines the maximum number of bytes in the data field the SONIC can transmit and receive packets up to 64k bytes in length.

### 2.6 FCS FIELD

The Frame Check Sequence (FCS) is a 32-bit CRC field calculated and appended to a packet during transmission to allow detection of error-free packets. During reception, an error-free packet results in a specific pattern in the CRC

generator. The AUTODIN II ( $X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X^1 + 1$ ) polynomial is used for the CRC calculations. The SONIC may optionally append the CRC sequence during transmission, and checks the CRC both during normal reception and self-reception during a transmission (see Section 1.2.1).

### 2.7 MAC (MEDIA ACCESS CONTROL) CONFORMANCE

The SONIC is designed to be compliant to the IEEE 802.3 MAC Conformance specification. The SONIC implements most MAC functions in silicon and provides hooks for the user software to handle the remaining functions. The MAC Conformance specifications are summarized in Table 2-1.

**TABLE 2-1. MAC Conformance Specifications**

Conformance Test Name	Support By		
	SONIC	User Driver Software	Notes
Minimum Frame Size	X		
Maximum Frame Size	X	X	1
Address Generation	X	X	2
Address Recognition	X		
Pad Length Generation	X	X	3
Start Of Frame Delimiter	X		
Length Field	X		
Preamble Generation	X		
Order of Bit Transmission	X		
Inconsistent Frame Length	X	X	1
Non-Integral Octet Count	X		
Incorrect Frame Check Sequence	X		
Frame Assembly	X		
FCS Generation and Insertion	X		
Carrier Deference	X		
Interframe Spacing	X		
Collision Detection	X		
Collision Handling	X		
Collision Backoff and Retransmission	X		
FCS Validation	X		
Frame Disassembly	X		
Back-to-Back Frames	X		
Flow Control	X		
Attempt Limit	X		
Jam Size (after SFD)	X		
Jam Size (in Preamble)	X		

**Note 1:** The SONIC provides the byte count of the entire packet in the RXpkt.byte\_count (see Section 3.4.3). The user's driver software may perform further filtering of the packet based upon the byte count.

**Note 2:** The SONIC does not provide Source Address insertion; however, a transmit descriptor fragment, containing only the Source Address, may be created for each packet. See Section 3.5.1.

**Note 3:** The SONIC does not provide Pad generation; however, the user's driver software can easily append the Pad by lengthening the TXpkt.pkt\_size field and TXpkt.frag\_size field(s) to at least 64 bytes. See Section 3.5.1.

### 3.0 Buffer Management

#### 3.1 BUFFER MANAGEMENT OVERVIEW

The SONIC's buffer management scheme is based on separate buffers and descriptors (*Figures 3-2 and 3-11*). Packets that are received or transmitted are placed in buffers called the Receive Buffer Area (RBA) and the Transmit Buffer Area (TBA). The system keeps track of packets in these buffers using the information in the Receive Descriptor Area (RDA) and the Transmit Descriptor Area (TDA). A single (TDA) points to a single TBA, but multiple RDAs can point to a single RBA (one RDA per packet in the buffer). The Receive Resource Area (RRA), which is another form of descriptor, is used to keep track of the actual buffer.

When packets are transmitted, the system sets up the packets in one or more TBAs with a TDA pointing to each TBA. There can only be one packet per TBA/TDA pair. A single TBA, however, may be made up of several fragments of data dispersed in memory. There is one TDA pointing to each TBA which specifies information about the buffer's size, location in memory, number of fragments and status after transmission. The TDAs are linked together in a linked list. The system causes the SONIC to transmit the packets by passing the first TDA to the SONIC and issuing the transmit command.

Before a packet can be received, an RDA and RBA must be set up by the system. RDA's are made up as a linked list similar to TDAs. An RDA is not linked to a particular RBA, though. Instead, an RDA is linked specifically to a packet after it has been buffered into an RBA. More than one packet can be buffered into the same RBA, but each packet gets its own RDA. A received packet can not be scattered into fragments. The system only needs to tell the SONIC where the first RDA and where the RBAs are. Since an RDA never specifically points to an RBA, the RRA is used to keep track of the RBAs. The RRA is a circular queue of pointers and buffer sizes (not a linked list). When the SONIC receives a packet, it is buffered into a RBA with a corresponding and unique RDA that is written to so that it points to and describes the new packet. If the RBA does not have enough space to buffer the next packet, a new RBA is obtained from the RRA.

#### 3.2 DESCRIPTOR AREAS

Descriptors are the basis of the buffer management scheme used by the SONIC. A RDA points to a received packet within a RBA, RRA points to a RBA and a TDA points to a TBA which contains a packet to be transmitted. The conventions and registers used to describe these descriptors are discussed in the next three sections.

##### 3.2.1 Naming Convention for Descriptors

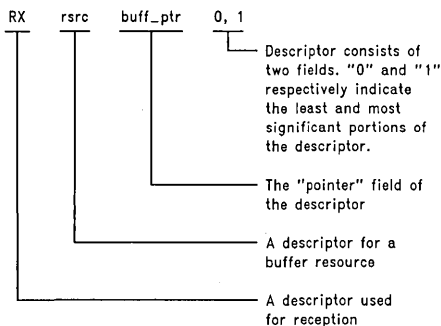
The fields which make up the descriptors are named in a consistent manner to assist in remembering the usage of each descriptor. Each descriptor name consists of three components in the following format.

[RX/TX][descriptor name].[field]

The first two capital letters indicate whether the descriptor is used for transmission (TX) or reception (RX), and is then followed by the descriptor name having one of two names.

rsrc = Resource descriptor  
pkt = Packet descriptor

The last component consists of a field name to distinguish it from the other fields of a descriptor. The field name is separated from the descriptor name by a period ("."). An example of a descriptor is shown below.



TL/F/10492-86

##### 3.2.2 Abbreviations

The abbreviations in Table 3.1 are used to describe the SONIC registers and data structures in memory. The "0" and "1" in the abbreviations indicate the least and most significant portions of the registers or descriptors. Table 3-1 lists the naming convention abbreviations for descriptors.

##### 3.2.3 Buffer Management Base Addresses

The SONIC uses three areas in memory to store descriptor information: the Transmit Descriptor Area (TDA), Receive Descriptor Area (RDA), and the Receive Resource Area (RRA). The SONIC accesses these areas by concatenating a 16-bit base address register with a 16-bit offset register. The base address register supplies a fixed upper 16 bits of address and the offset registers provide the lower 16 bits of address. The base address registers are the Upper Transmit Descriptor Address (UTDA), Upper Receive Descriptor Address (URDA), and the Upper Receive Resource Address (URRA) registers. The corresponding offset registers are shown below.

Upper Address Registers	Offset Registers
URRA	RSA, REA, RWP, RRP
URDA	GRDA
UTDA	CTDA

See Table 3-1 for definition of register mnemonics.

*Figure 3-1* shows an example of the Transmit Descriptor Area and the Receive Descriptor Area being located by the UTDA and URDA registers. The descriptor areas, RDA, TDA, and RRA are allowed to have the same base address. i.e., URRA=URDA=UTDA. Care, however, must be taken to prevent these areas from overwriting each other.



### 3.0 Buffer Management (Continued)

TABLE 3-1. Descriptor Abbreviations

TRANSMIT AND RECEIVE AREAS	
RRA	Receive Resource Area
RDA	Receive Descriptor Area
RBA	Receive Buffer Area
TDA	Transmit Descriptor Area
TBA	Transmit Buffer Area
BUFFER MANAGEMENT REGISTERS	
RSA	Resource Start Area Register
REA	Resource End Area Register
RRP	Resource Read Pointer Register
RWP	Resource Write Pointer Register
CRDA	Current Receive Descriptor Address Register
CRBA0,1	Current Receive Buffer Address Register
TCBA0,1	Temporary Current Buffer Address Register
RBWC0,1	Remaining Buffer Word Count Register
TRBWC0,1	Temporary Remaining Buffer Word Count Register
EOBC	End of Buffer Count Register
TPS	Transmit Packet Size Register
TSA0,1	Transmit Start Address Register
CTDA	Current Transmit Descriptor Address Register

BUFFER MANAGEMENT REGISTERS (Continued)	
TFC	Transmit Fragment Count Register
TFS	Transmit Fragment Size Register
UTDA	Upper Transmit Descriptor Address Register
URRA	Upper Receive Resource Address Register
URDA	Upper Receive Descriptor Address Register
TRANSMIT AND RECEIVE DESCRIPTORS	
RXsrc.buff_ptr0,1	Buffer Pointer Field in the RRA
RXsrc.buff_wc0,1	Buffer Word Count Fields in the RRA
RXpkt.status	Receive Status Field in the RDA
RXpkt.byte_count	Packet Byte Count Field in the RDA
RXpkt.buff_ptr0,1	Buffer Pointer Fields in the RDA
RXpkt.link	Receive Descriptor Link Field in RDA
RXpkt.in_use	"In Use" Field in RDA
TXpkt.frag_count	Fragment Count Field in TDA
TXpkt.pkt_size	Packet Size Field in TDA
TXpkt.pkt_ptr0,1	Packet Pointer Fields in TDA
TXpkt.frag_size	Fragment Size Field in TDA
TXpkt.link	Transmit Descriptor Link Field in TDA

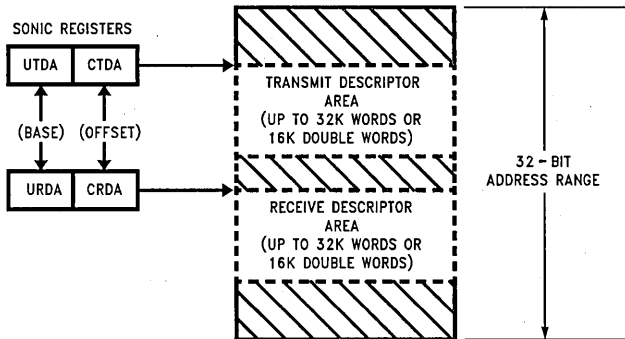


FIGURE 3-1. Transmit and Receive Descriptor Pointers

TL/F/10492-10



### 3.0 Buffer Management (Continued)

#### 3.3 DESCRIPTOR DATA ALIGNMENT

All fields used by descriptors (RXpkt.xxx, RXsrc.xxx, and TXpkt.xxx) are word quantities (16-bit) and must be aligned to word boundaries (A0=0) for 16-bit memory and to long word boundaries (A1,A0=0,0) for 32-bit memory. The Receive Buffer Area (RBA) must also be aligned to a word boundary in 16-bit mode and a long word boundary in 32-bit mode. The fragments in the Transmit Buffer Area (TBA), however, may be aligned on any arbitrary byte boundary.

#### 3.4 RECEIVE BUFFER MANAGEMENT

The Receive Buffer Management operates on three areas in memory into which data, status, and control information are written during reception (Figure 3-2). These three areas must be initialized (Section 3.4.4) before enabling the receiver (setting the RXEN bit in the Command register). The receive resource area (RRA) contains descriptors that locate receive buffer areas in system memory. These descriptors are denoted by R1, R2, etc. in Figure 3-2. Packets (denoted by P1, P2, etc.) can then be buffered into the corresponding RBAs. Depending on the size of each buffer area and the size of the packet(s), multiple or single packets are buffered into each RBA. The receive descriptor area (RDA) contains status and control information for each packet (D1, D2, etc. in Figure 3-2) corresponding to each received packet (D1 goes with P1, D2 with P2, etc.).

When a packet arrives, the address recognition logic checks the address for a Physical, Multicast, or Broadcast match and if the packet is accepted, the SONIC buffers the packet contiguously into the selected Receive Buffer Area (RBA). Because of the previous end-of-packet processing, the SONIC assures that the complete packet is written into a single contiguous block. When the packet ends, the SONIC writes the receive status, byte count, and location of the packet into the Receive Descriptor Area (RDA). The SONIC then updates its pointers to locate the next available descriptor and checks the remaining words available in the RBA. If sufficient space remains, the SONIC buffers the next packet immediately after the previous packet. If the current buffer is out of space the SONIC fetches a Resource descriptor from the Receive Resource Area (RRA) acquiring an additional buffer that has been previously allocated by the system.

#### 3.4.1 Receive Resource Area (RRA)

As buffer memory is consumed by the SONIC for storing data, the Receive Resource Area (RRA) provides a mechanism that allows the system to allocate additional buffer space for the SONIC. The system loads this area with resource descriptors that the SONIC, in turn, reads as its current buffer space is used up. Each resource descriptor consists of a 32-bit buffer pointer locating the starting point of the RBA and a 32-bit Word Count that indicates the size of the buffer in words (2 bytes per word). The buffer pointer and word count are contiguously located using the format shown in Figure 3-3 with each component composed of 16-bit fields. The SONIC stores this information internally and concatenates the corresponding fields to create 32-bit long words for the buffer pointer and word count. Note that in 32-bit mode the upper word (D<31:16>) is not used by the SONIC. This area may be used for other purposes since the SONIC never writes into the RRA.

The SONIC organizes the RRA as a circular queue for efficient processing of descriptors. Four registers define the RRA. The first two, the Resource Start Area (RSA) and the Resource End Area (REA) registers, determine the starting and ending locations of the RRA, and the other two registers update the RRA. The system adds descriptors at the address specified by the Resource Write Pointer (RWP), and the SONIC reads the next descriptor designated by the Resource Read Pointer (RRP). The RRP is advanced 4 words in 16-bit mode (4 long words in 32-bit mode) after the SONIC finishes reading the RRA and automatically wraps around to the beginning of the RRA once the end has been reached. When a descriptor in the RRA is read, the RXsrc.buf\_pt0,1 is loaded into the CRBA0,1 registers and the RXsrc.buf\_wc0,1 is loaded into the RBWC0,1 registers.

The alignment of the RRA is confined to either word or long word boundaries, depending upon the data width mode. In 16-bit mode, the RRA must be aligned to a word boundary (A0 is always zero) and in 32-bit mode, the RRA is aligned to a long word boundary (A0 and A1 are always zero).

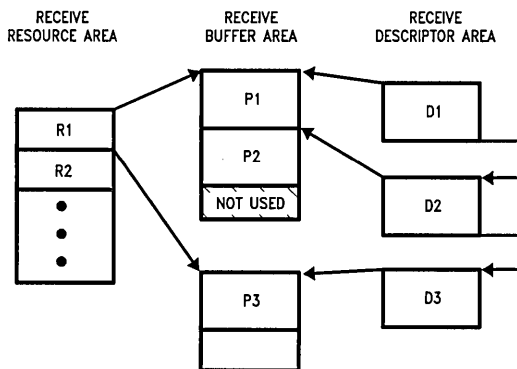


FIGURE 3-2. Overview of Receive Buffer Management

TL/F/10492-11

### 3.0 Buffer Management (Continued)

#### 3.4.2 Receive Buffer Area (RBA)

The SONIC stores the actual data of a received packet in the RBA. The RBAs are designated by the resource descriptors in the RRA as described above. The `RXsrc.buf_wc0,1` fields of the RRA indicate the length of the RBA. When the SONIC gets a RBA from the RRA, the `RXsrc.buf_wc0,1` values are loaded into the Remaining Buffer Word Count registers (`RBWC0,1`). These registers keep track of how much space (in words) is left in the buffer. When a packet is buffered in a RBA, it is buffered contiguously (the SONIC will not scatter a packet into multiple buffers or fragments). Therefore, if there is not enough space left in a RBA after buffering a packet to buffer at least one more maximum sized packet (the maximum legal sized packet expected to be received from the network), a new buffer must be acquired. The End of Buffer Count (EOBC) register is used to tell the SONIC the maximum packet size that the SONIC will need to buffer.

##### 3.4.2.1 End of Buffer Count (EOBC)

The EOBC is a boundary in the RBA based from the bottom of the buffer. The value written into the EOBC is the maximum expected size (in words) of the network packet that the SONIC will have to buffer. This word count creates a line in the RBA that, when crossed, causes the SONIC to fetch a new RBA resource from the RRA.

**Note:** The EOBC is a word count, not a byte count. Also, the value programmed into EOBC must be a double word (32-bit) quantity when the SONIC is in 32-bit mode (e.g. in 32-bit mode, EOBC should be set to 758 words, not 759 words even though the maximum size of an IEEE 802.3 packet is 759 words).

##### 3.4.2.2 Buffering the Last Packet In an RBA

At the start of reception, the SONIC stores the packet beginning at the Current Receive Buffer Address (`CRBA0,1`) and continues until the reception is complete. Concurrent with reception, the SONIC decrements the Remaining Buffer Word Count (`RBWC0,1`) by one in 16-bit mode or by two in 32-bit mode. At the end of reception, if the packet has crossed the EOBC boundary, the SONIC knows that the next packet might not fit in the RBA. This check is done by comparing the `RBWC0,1` registers with the EOBC. If `RBWC0,1` is less than the EOBC (the last packet buffered has crossed the EOBC boundary), the SONIC fetches the next resource descriptor in the RRA. If `RBWC0,1` is greater than or equal to the EOBC (the EOBC boundary has not been crossed) the next packet reception continues at the present location pointed to by `CRBA0,1` in the same RBA. *Figure 3-4* illustrates the SONIC's actions for (1)  $RBWC0,1 \geq EOBC$  and (2)  $RBWC0,1 < EOBC$ . See Section 3.4.4.4 for specific information about setting the EOBC.

**Note:** It is important that the EOBC boundary be "crossed." In other words, case #1 in *Figure 3-4* must exist before case #2 exists. If case #2 occurs without case #1 having occurred first, the test for  $RBWC0,1 < EOBC$  will not work properly and the SONIC will not fetch a new buffer. The result of this will be a buffer overflow (RBAE in the Interrupt Status Register, Section 4.3.6).

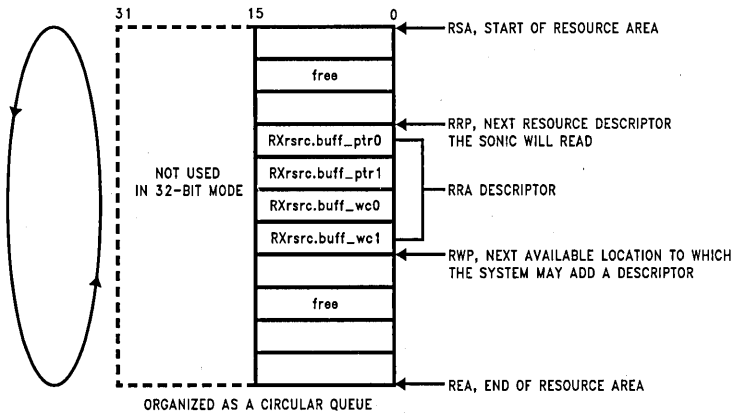
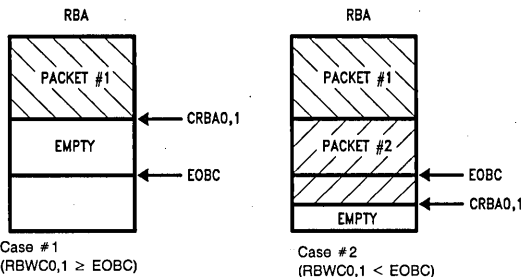


FIGURE 3-3. Receive Resource Area Format

TL/F/10492-12



TL/F/10492-13

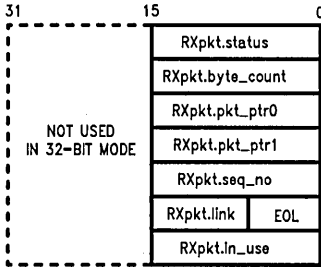
Case #1: SONIC buffers next packet in same RBA.  
Case #2: SONIC detects an exhausted RBA and will buffer the next packet in another RBA.

FIGURE 3-4. Receive Buffer Area

### 3.0 Buffer Management (Continued)

#### 3.4.3 Receive Descriptor Area (RDA)

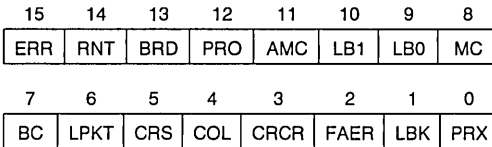
After the SONIC buffers a packet to memory, it writes 6 words of status and control information into the RDA, reads the link field to the next receive descriptor, and writes to the in-use field of the current descriptor. In 32-bit mode, the upper word, D<31:16>, is not used. This unused area in memory should not be used for other purposes, since the SONIC may still write into these locations. Each receive descriptor consists of the following sections (*Figure 3-5*).



TL/F/10492-14

**FIGURE 3-5. Receive Descriptor Format**

**receive status:** indicates status of the received packet. The SONIC writes the Receive Control register into this field. *Figure 3-6* shows the receive status format. This field is loaded from the contents of the Receive Control register. Note that ERR, RNT, BRD, PRO, and AMC are configuration bits and are programmed during initialization. See Section 4.3.3 for the description of the Receive Control register.

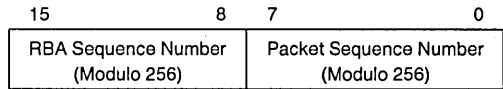


**FIGURE 3-6. Receive Status Format**

**byte count:** gives the length of the complete packet from the start of Destination Address to the end of FCS.

**packet pointer:** a 32-bit pointer that locates the packet in the RBA. The SONIC writes the contents of the CRBA0,1 registers into this field.

**sequence numbers:** this field displays the contents of two 8-bit counters (modulo 256) that sequence the RBAs used and the packets buffered. These counters assist the system in determining when an RBA has been completely processed. The sequence numbers allow the system to tally the packets that have been processed within a particular RBA. There are two sequence numbers that describe a packet: the RBA Sequence Number and the Packet Sequence Number. When a packet is buffered to memory, the SONIC maintains a single RBA Sequence Number for all packets in an RBA and sequences the Packet Number for succeeding packets in the RBA. When the SONIC uses the next RBA, it increments the RBA Sequence Number and clears the Packet Sequence Number. The RBA's sequence counter is not incremented when the read RRA command is issued in the Command register. The format of the Receive Sequence Numbers are shown in *Figure 3-7*. These counters are reset during hardware reset or by writing zero to them.



**FIGURE 3-7. Receive Sequence Number Format**

**receive link field:** a 15-bit pointer (A15-A1) that locates the next receive descriptor. The LSB of this field is the End Of List (EOL) bit, and indicates the last descriptor in the list. (Initialized by the system.)

**In use field:** this field provides a handshake between the system and the SONIC to indicate the ownership of the descriptor. When the system avails a descriptor to the SONIC, it writes a non-zero value into this field. The SONIC, in turn, sets this field to all "0's" when it has finished processing the descriptor. (That is, when the CRDA register has advanced to the next receive descriptor.) Generally, the SONIC releases control after writing the status and control information into the RDA. If, however, the SONIC has reached the last descriptor in the list, it maintains ownership of the descriptor until the system has appended additional descriptors to the list. The SONIC then relinquishes control after receiving the next packet. (See Section 3.4.6.1 for details on when the SONIC writes to this field). The receive packet descriptor format is shown in *Figure 3-5*.

#### 3.4.4 Receive Buffer Management Initialization

The Receive Resource, Descriptor, and Buffer areas (RRA, RDA, RBA) in memory and the appropriate SONIC registers must be properly initialized before the SONIC begins buffering packets. This section describes the initialization process.

##### 3.4.4.1 Initializing The Descriptor Page

All descriptor areas (RRA, RDA, and TDA) used by the SONIC reside within areas up to 32k (word) or 16k (long word) pages. This page may be placed anywhere within the 32-bit address range by loading the upper 16 address lines into the UTDA, URDA, and URRR registers.

##### 3.4.4.2 Initializing The RRA

The initialization of the RRA consists of loading the four SONIC RRA registers and writing the resource descriptor information to memory.

The RRA registers are loaded with the following values.

**Resource Start Area (RSA) register:** The RSA is loaded with the lower 16-bit address of the beginning of the RRA.

**Resource End Area (REA) register:** The REA is loaded with the lower 16-bit address of the end of the RRA. The end of the RRA is defined as the address of the last RXrsrc.ptr0 field in the RRA plus 4 words in 16-bit mode or 4 long words in 32-bit mode (*Figure 3-3*).

**Resource Read Pointer (RRP) register:** The RRP is loaded with the lower 16-bit address of the first resource descriptor the SONIC reads.

**Resource Write Pointer (RWP) register:** The RWP is loaded with the lower 16-bit address of the next vacant location where a resource descriptor will be placed by the system.

**Note:** The RWP register must only point to either (1) the RXrsrc.ptr0 field of one of the RRA Descriptors, (2) the memory address that the RSA points to (the start of the RRA), or (3) the memory address that the REA points to (the end of the RRA). When the RWP = RRP comparison is made, it is performed after the complete RRA descriptor has been read and not during the fetch. Failure to set the RWP to any of the above values prevents the RWP = RRP comparison from ever becoming true.

### 3.0 Buffer Management (Continued)

All RRA registers are concatenated with the URRRA register for generating the full 32-bit address.

The resource descriptors that the system writes to the RRA consists of four fields: (1) RXrsrc.buff\_ptr0, (2) RXrsrc.buff\_ptr1, (3) RXrsrc.buff\_wc0, and (4) RXrsrc.buff\_wc1. The fields must be contiguous (they cannot straddle the end points) and are written in the order shown in Figure 3-8. The "0" and "1" in the descriptors denote the least and most significant portions for the Buffer Pointer and Word Count. The first two fields supply the 32-bit starting location of the Receive Buffer Area (RBA), and the second two define the number of 16-bit words that the RBA occupies.

Note that two restrictions apply to the Buffer Pointer and Word Count. First, in 32-bit mode, since the SONIC always writes long words, an even count must be written to RXrsrc.buff\_wc0. Second, the Buffer Pointer must either be pointing to a word boundary in 16-bit mode (A0=0) or a long word boundary in 32-bit mode (A0,A1=0,0). Note also that the descriptors must be properly aligned in the RRA as discussed in Section 3.3.

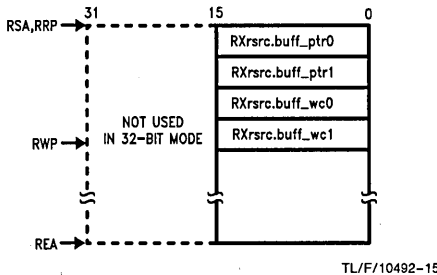


FIGURE 3-8. RRA Initialization

TL/F/10492-15

After configuring the RRA, the RRA Read command (setting RRA bit in the Command register) may be given. This command causes the SONIC to read the RRA descriptor in a single block operation, and load the following registers (see Section 4.2 for register mnemonics):

- CRBA0 register ← RXrsrc.buff\_ptr0
- CRBA1 register ← RXrsrc.buff\_ptr1
- RBWC0 register ← RXrsrc.buff\_wc0
- RBWC1 register ← RXrsrc.buff\_wc1

When the command has completed, the RRA bit in the Command register is reset to "0". Generally this command is only issued during initialization. At all other times, the RRA is automatically read as the SONIC finishes using an RBA.

#### 3.4.4.3 Initializing The RDA

To accept multiple packets from the network, the receive packet descriptors must be linked together via the RXpkt.link fields. Each link field must be written with a 15-bit (A15-A1) pointer to locate the beginning of the next descriptor in the list. The LSB of the RXpkt.link field is the End of List (EOL) bit and is used to indicate the end of the descriptor list. EOL = 1 for the last descriptor and EOL = 0 for the first or middle descriptors. The RXpkt.in\_use field indicates whether the descriptor is owned by the SONIC. The system writes a non-zero value to this field when the descriptor is available, and the SONIC writes all "0's" when it finishes using the descriptor. At startup, the Current Receive Descriptor Address (CRDA) register must be loaded with the address of the first RXpkt.status field in order for

the SONIC to begin receive processing at the first descriptor. An example of two descriptors linked together is shown in Figure 3-9. The fields initialized by the system are displayed in **bold type**. The other fields are written by the SONIC after a packet is accepted. The RXpkt.in\_use field is first written by the system, and then by the SONIC. Note that the descriptors must be aligned properly as discussed in Section 3.3. Also note that the URDA register is concatenated with the CRDA register to generate the full 32-bit address.

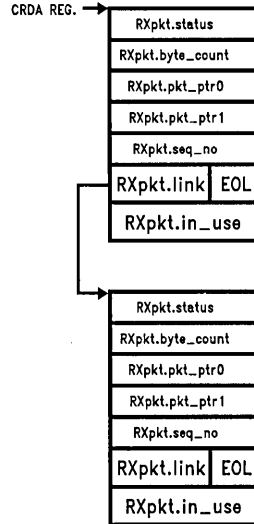


FIGURE 3-9. RDA Initialization Example

TL/F/10492-16

#### 3.4.4.4 Initializing the Lower Boundary of the RBA

A "false bottom" is set in the RBA by loading the End Of Buffer Count (EOBC) register with a value equal to the maximum size packet in words (16 bits) that may be received. This creates a lower boundary in the RBA. Whenever the Remaining Buffer Word Count (RBWC0,1) registers decrement below the EOBC register, the SONIC buffers the next packet into another RBA. This also guarantees that a packet is always contiguously buffered into a single Receive Buffer Area (RBA). The SONIC does not buffer a packet into multiple RBAs. Note that in 32-bit mode, the SONIC holds the LSB always low so that it properly compares with the RBWC0,1 registers.

After a hardware reset, the EOBC reset, the EOBC register is automatically initialized to 2F8h (760 words or 1520 bytes). For 32-bit applications this is the suggested value for EOBC. EOBC defaults to 760 words (1520 bytes) instead of 759 words (1518 bytes) because 1518 is not a double word (32-bit) boundary (see Section 3.4.2.1). If the SONIC is used in 16-bit mode, then EOBC should be set to 759 words (1518 bytes) because 1518 is a word (16-bit) boundary.

Sometimes it may be desired to buffer a single packet per RBA. When doing this, it is important to set EOBC and the buffer size correctly. The suggested practice is to set EOBC to a value that is at least 4 bytes, in 32-bit mode, or 2 bytes, in 16-bit mode, less than the buffer size. An example of this for 32-bit mode is to set EOBC to 760 words (1520 bytes)

### 3.0 Buffer Management (Continued)

and the buffer size to 762 words (1524 bytes). A similar example for 16-bit mode would be EOBC = 759 words (1518 bytes) and the buffer size set to 760 words (1520 bytes). The buffer can be any size, but as long as the EOBC is 2 words, for 32-bit mode, or 1 word, for 16-bit mode, less than the buffer size, only one packet will be buffered in that RBA.

**Note 1:** It is possible to filter out most oversized packets by setting the buffer size to 760 words (1520 bytes) in 32-bit mode or 759 words (1518 bytes) in 16-bit mode. EOBC would be set to 758 words (1516 bytes) for both cases. With this configuration, any packet over 1520 bytes, in 32-bit mode, or 1518 bytes, in 16-bit mode, will not be completely buffered because the packet will overflow the buffer. When a packet overflow occurs, a Receive Buffer Area Exceeded interrupt (RBAE) in the Interrupt Status Register, Section 4.3.6) will occur.

**Note 2:** When buffering one packet per buffer, it is suggested that the values in Note 1 above be used. Since the minimum legal sized Ethernet packet is 64 bytes, however, it is possible to set EOBC as much as 64 bytes less than the buffer size and still end up with one packet per buffer. Figure 3-10 shows this "range."

#### 3.4.5 Beginning Of Reception

At the beginning of reception, the SONIC checks its internally stored EOL bit from the previous RXpkt.link field for a "1". If the SONIC finds EOL = 1, it recognizes that after the previous reception, there were no more remaining receive packet descriptors. It re-reads the same RXpkt.link field to check if the system has updated this field since the last reception. If the SONIC still finds EOL = 1, reception ceases. (See Section 3.5 for adding descriptors to the list.) Otherwise, the SONIC begins storing the packet in the RBA starting at the Current Receive Buffer Address (CRBA0,1) registers and continues until the packet has completed. Concurrent with the packet reception, the Remaining Buffer Word Count (RBWC0,1) registers are decremented after each word is written to memory. This register determines the remaining words in the RBA at the end of reception.

#### 3.4.6 End Of Packet Processing

At the end of a reception, the SONIC enters its end of packet processing sequence to determine whether to accept or reject the packet based on receive errors and packet size. At the end of reception the SONIC enters one of the following two sequences:

- Successful reception sequence
- Buffer recovery for runt packets or packets with errors

#### 3.4.6.1 Successful Reception

If the SONIC accepts the packet, it first writes 5 words of descriptor information in the RDA beginning at the address pointed to by the Current Receive Descriptor Address (CRDA) register. It then reads the RXpkt.link field to advance the CRDA register to the next receive descriptor. The SONIC also checks the EOL bit for a "1" in this field. If EOL = 1, no more descriptors are available for the SONIC. The SONIC recovers the address of the current RXpkt.link field (from a temporary register) and generates a "Receive Descriptors Exhausted" indication in the Interrupt Status register. (See Section 3.4.7 on how to add descriptors.) The SONIC maintains ownership of the descriptor by not writing to the RXpkt.in\_\_use field. Otherwise, if EOL = 0, the SONIC advances the CRDA register to the next descriptor and resets the RXpkt.in\_\_use field to all "0's".

The SONIC accesses the complete 7 word RDA descriptor in a single block operation.

The SONIC also checks if there is remaining space in the RBA. The SONIC compares the Remaining Buffer Word Count (RBWC0,1) registers with the static End Of Buffer Count (EOBC). If the RBWC is less than the EOBC, a maximum sized packet will no longer fit in the remaining space in the RBA; hence, the SONIC fetches a resource descriptor from the RRA and loads its registers with the pointer and word count of the next available RBA.

#### 3.4.6.2 Buffer Recovery For Runt Packets Or Packets With Errors

If a runt packet (less than 64 bytes) or packet with errors arrives and the Receive Control register has been configured to not accept these packets, the SONIC recovers its pointers back to the original positions. The CRBA0,1 registers are not advanced and the RBWC0,1 registers are not decremented. The SONIC recovers its pointers by maintaining a copy of the buffer address in the Temporary Receive Buffer Address registers (TRBA0,1). The SONIC recovers the value in the RBWC0,1 registers from the Temporary Buffer Word Count registers (TBWC0,1).

#### 3.4.7 Overflow Conditions

When an overflow condition occurs, the SONIC halts its DMA operations to prevent writing into unauthorized memory. The SONIC uses the Interrupt Status register (ISR) to indicate three possible overflow conditions that can occur

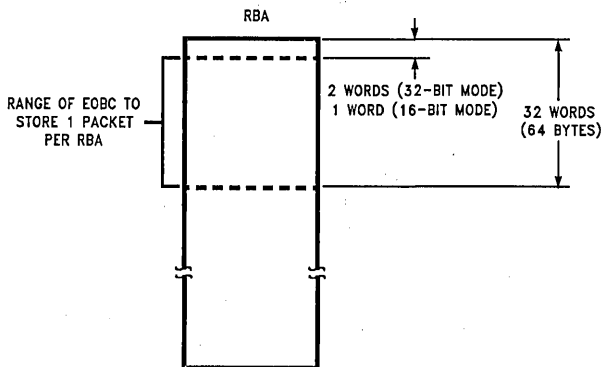


FIGURE 3-10. Setting EOBC for Single Packet RBA

TL/F/10492-17

### 3.0 Buffer Management (Continued)

when its receive resources have been exhausted. The system should respond by replenishing the resources that have been exhausted. These overflow conditions (Descriptor Resources Exhausted, Buffer Resources Exhausted, and RBA Limit Exceeded) are indicated in the Interrupt Status register and are detailed as follows:

**Descriptor Resources Exhausted:** This occurs when the SONIC has reached the last receive descriptor in the list, meaning that the SONIC has detected  $EOL = 1$ . The system must supply additional descriptors for continued reception. The system can do this in one of two ways: 1) appending descriptors to the existing list, or 2) creating a separate list.

1) Appending descriptors to the existing list. This is the easiest and preferred way. To do this, the system, after creating the new list, joins the new list to the existing list by simply writing the beginning address of the new list into the `RXpkt.link` field and setting  $EOL = 0$ . At the next reception, the SONIC re-reads the last `RXpkt.link` field, and updates its CRDA register to point to the next descriptor.

2) Creating a separate list. This requires an additional step because the lists are not joined together and requires that the CRDA register be loaded with the address of the `RXpkt.link` field in the new list.

During this overflow condition, the SONIC maintains ownership of the descriptor (`RXpkt.in_use`  $\neq$  00h) and waits for the system to add additional descriptors to the list. When the system appends more descriptors, the SONIC releases ownership of the descriptor after writing 0000h to the `RXpkt.in_use` field.

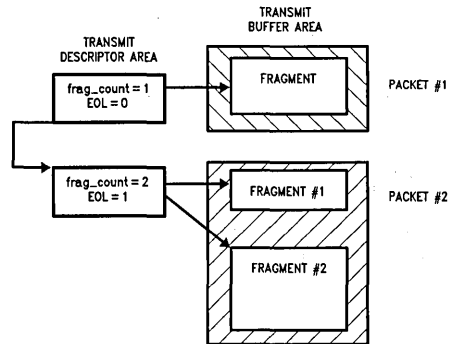
**Buffer Resources Exhausted:** This occurs when the SONIC has detected that the Resource Read Pointer (RRP) and Resource Write Pointer (RWP) registers are equal (i.e., all RRA descriptors have been exhausted). The RBE bit in the Interrupt Status register is set when the SONIC finishes using the second to last receive buffer and reads the last RRA descriptor. Actually, the SONIC is not truly out of resources, but gives the system an early warning of an impending out of resources condition. To continue reception after the last RBA is used, the system must supply additional RRA descriptor(s), update the RWP register, and clear the RBE bit in the ISR. The SONIC rereads the RRA after this bit is cleared.

**RBA Limit Exceeded:** This occurs when a packet does not completely fit within the remaining space of the RBA. This can occur if the EOBC register is not programmed to a value greater than the largest packet that can be received. When this situation occurs, the packet is truncated and the SONIC reads the RRA to obtain another RBA. Indication of an RBA limit being exceeded is signified by the Receive Buffer Area Exceeded (RBAE) interrupt being set (see Section 4.3.6). An RDA will not be set up for the truncated packet and the buffer space will not be re-used. To rectify this potential overflow condition, the EOBC register must be loaded with a value equal to or greater than the largest packet that can be accepted. (See Section 3.4.2.)

#### 3.5 TRANSMIT BUFFER MANAGEMENT

To begin transmission, the system software issues the Transmit command ( $TXP = 1$  in the CR). The Transmit Buffer Management uses two areas in memory for transmitting packets (Figure 3-11), the Transmit Descriptor Area (TDA)

and the Transmit Buffer Area (TBA). During transmission, the SONIC fetches control information from the TDA, loads its appropriate registers, and then transmits the data from the TBA. When the transmission is complete, the SONIC writes the status information in the TDA. From a single transmit command, packets can either be transmitted singly or in groups if several descriptors have been linked together.



TL/F/10492-18

**FIGURE 3-11. Overview of Transmit Buffer Management**

#### 3.5.1 Transmit Descriptor Area (TDA)

The TDA contains descriptors that the system has generated to exchange status and control information. Each descriptor corresponds to a single packet and consists of the following 16-bit fields.

**TXpkt.status:** This field is written by the SONIC and provides status of the transmitted packet. (See Section 3.5.1.2 for more details.)

**TXpkt.config:** This field allows programming the SONIC to one of the various transmit modes. The SONIC reads this field and loads the corresponding configuration bits (PINT, POWC, CRCI, and EXDIS) into the Transmit Control register. (See Section 3.5.1.1 for more details.)

**TXpkt.pkt\_size:** This field contains the byte count of the entire packet.

**TXpkt.frag\_count:** This field contains the number of fragments the packet is segmented into.

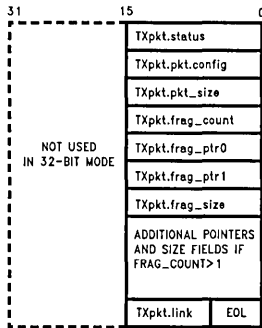
**TXpkt.frag\_ptr0,1:** This field contains a 32-bit pointer which locates the packet fragment to be transmitted in the Transmit Buffer Area (TBA). This pointer is not restricted to any byte alignment.

**TXpkt.frag\_size:** This field contains the byte count of the packet fragment. The minimum fragment size is 1 byte.

**TXpkt.link:** This field contains a 15-bit pointer (A15-A1) to the next TDA descriptor. The LSB, the End Of List (EOL) bit, indicates the last descriptor in the list when set to a "1". When descriptors have been linked together, the SONIC transmits back-to-back packets from a single transmit command.

The data of the packet does not need to be contiguous, but can exist in several locations (fragments) in memory. In this case, the `TXpkt.frag_count` field is greater than one, and additional `TXpkt.frag_ptr0,1` and `TXpkt.frag_size` fields corresponding to each fragment are used. The descriptor format is shown in Figure 3-12. Note that in 32-bit mode the upper word, `D<31:16>`, is not used.

### 3.0 Buffer Management (Continued)



TL/F/10492-19

FIGURE 3-12. Transmit Descriptor Area

#### 3.5.1.1 Transmit Configuration

The TXpkt.config field allows the SONIC to be programmed into one of the transmit modes before each transmission. At the beginning of each transmission, the SONIC reads this field and loads the PINT, POWC, CRCI, and EXDIS bits into the Transmit Control register (TCR). The configuration bits in the TCR correspond directly with the bits in the TXpkt.config field as shown in Figure 3-13. See Section 4.3.4 for the description on the TCR.

15	14	13	12	11	10	9	8
PINT	POWC	CRCI	EXDIS	X	X	X	X
7	6	5	4	3	2	1	0
X	X	X	X	X	X	X	X

Note: x = don't care

FIGURE 3-13. TXpkt.config Field

#### 3.5.1.2 Transmit Status

At the end of each transmission the SONIC writes the status bits (<10:0>) of the Transmit Control Register (TCR) and the number of collisions experienced during the transmission into the TXpkt.status field (Figure 3-14, res = reserved). Bits NC4-NC0 indicate the number of collisions where NC4 is the MSB. See Section 4.3.4 for the description of the TCR.

15	14	13	12	11	10	9	8
NC4	NC3	NC2	NC1	NC0	EXD	DEF	NCRS
7	6	5	4	3	2	1	0
CRSL	EXC	OWC	res	PMB	FU	BCM	PTX

FIGURE 3-14. TXpkt.status Field

#### 3.5.2 Transmit Buffer Area (TBA)

The TBA contains the fragments of packets that are defined by the descriptors in the TDA. A packet can consist of a single fragment or several fragments, depending upon the fragment count in the TDA descriptor. The fragments also can reside anywhere within the full 32-bit address range, and be aligned to any byte boundary. When an odd byte boundary is given, the SONIC automatically begins reading data at the corresponding word boundary in 16-bit mode or a long word boundary in 32-bit mode. The SONIC ignores the extraneous bytes which are written into the FIFO during

odd byte alignment fragments. The minimum allowed fragment size is 1 byte. Figure 3-11 shows the relationship between the TDA and the TBA for single and multi-fragmented packets.

#### 3.5.3 Preparing To Transmit

All fields in the TDA descriptor and the Current Transmit Descriptor Address (CTDA) register of the SONIC must be initialized before the Transmit Command (setting the TXP bit in the Command register) can be issued. If more than one packet is queued, the descriptors must be linked together with the TXpkt.link field. The last descriptor must have EOL = 1 and all other descriptors must have EOL = 0. To begin transmission, the system loads the address of the first TXpkt.status field into the CTDA register. Note that the upper 16-bits of address are loaded in the Upper Transmit Descriptor (UTDA) register. The user performs the following transmit initialization.

- 1) Initialize the TDA
- 2) Load the CTDA register with the address of the first transmit descriptor
- 3) Issue the transmit command

Note that if the Source Address of the packet being transmitted is not in the CAM, the Packet Monitored Bad (PMB) bit in the TXpkt.status field will be set (see Section 6.3.4).

#### 3.5.3.1 Transmit Process

When the Transmit Command (TXP = 1 in the Command register) is issued, the SONIC fetches the control information in the TDA descriptor, loads its appropriate registers (shown below) and begins transmission. (See Section 4.2 for register mnemonics.)

- TCR ← TXpkt.config
- TPS ← TXpkt.pkt\_size
- TFC ← TXpkt.frag\_count
- TSA0 ← TXpkt.frag\_ptr0
- TSA1 ← TXpkt.frag\_ptr1
- TFS ← TXpkt.frag\_size
- CTDA ← TXpkt.link

(CTDA is loaded after all fragments have been read and successfully transmitted. If the halt transmit command is issued (HTX bit in the Command register is set) the CTDA register is not loaded.)

During transmission, the SONIC reads the packet descriptor in the TDA and transmits the data from the TBA. If TXpkt.frag\_count is greater than one, the SONIC, after finishing transmission of the fragment, fetches the next TXpkt.frag\_ptr0,1 and TXpkt.frag\_size fields and transmits the next fragment. This process continues until all fragments of a packet are transmitted. At the end of packet transmission, status is written in to the TXpkt.status field. The SONIC then reads the TXpkt.link field and checks if EOL = 0. If it is "0", the SONIC fetches the next descriptor and transmits the next packet. If EOL = 1 the SONIC generates a "Transmission Done" indication in the Interrupt Status register and resets the TXP bit in the Command register.

In the event of a collision, the SONIC recovers its pointer in the TDA and retransmits the packet up to 15 times. The SONIC maintains a copy of the CTDA register in the Temporary Transmit Descriptor Address (TTDA) register.

The SONIC performs a block operation of 6, 3, or 2 accesses in the TDA, depending on where the SONIC is in the transmit process. For the first fragment, it reads the



### 3.0 Buffer Management (Continued)

TXpkt.config to TXpkt.frag\_size (6 accesses). For the next fragment, if any, it reads the next 3 fields from TXpkt.frag\_ptr0 to TXpkt.frag\_size (3 accesses). At the end of transmission it writes the status information to TXpkt.status and reads the TXpkt.link field (2 accesses).

#### 3.5.3.2 Transmit Completion

The SONIC stops transmitting under two conditions. In the normal case, the SONIC transmits the complete list of descriptors in the TDA and stops after it detects EOL = 1. In the second case, certain transmit errors cause the SONIC to abort transmission. If FIFO Underrun, Byte Count Mismatch, Excessive Collision, or Excessive Deferral (if enabled) errors occur, transmission ceases. The CTDA register points to the last packet transmitted. The system can also halt transmission under software control by setting the HTX bit in the Command register. Transmission halts after the SONIC writes to the TXpkt.status field.

#### 3.5.4 Dynamically Adding TDA Descriptors

Descriptors can be dynamically added during transmission without halting the SONIC. The SONIC can also be guaranteed to transmit the complete list including newly appended descriptors (barring any transmit abort conditions) by observing the following rule: The last TXpkt.link field must point to the next location where a descriptor will be added (see step 3 below and Figure 3-15). The procedure for appending descriptors consists of:

1. Creating a new descriptor with its TXpkt.link pointing to the next vacant descriptor location and its EOL bit set to a "1".
2. Resetting the EOL bit to a "0" of the previously last descriptor.
3. Re-issuing the Transmit command (setting the TXP bit in the Command register).

Step 3 assures that the SONIC will transmit all the packets in the list. If the SONIC is currently transmitting, the Transmit command has no effect and continues transmitting until it detects EOL = 1. If the SONIC had just finished transmitting, it continues transmitting from where it had previously stopped.

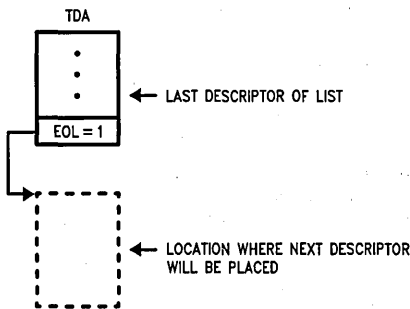


FIGURE 3-15. Initializing Last Link Field

TL/F/10492-20

### 4.0 SONIC Registers

The SONIC contains two sets of registers: The status/control registers and the CAM memory cells. The status/control registers are used to configure, control, and monitor SONIC operation. They are directly addressable registers and occupy 64 consecutive address locations in the system memory space (selected by the RA5-RA0 address pins). There are a total of 64 status/control registers divided into the following categories:

**User Registers:** These registers are accessed by the user to configure, control, and monitor SONIC operation. These are the only SONIC registers the user needs to access. Figure 4-3 shows the programmer's model and Table 4-1 lists the attributes of each register.

**Internal Use Registers:** These registers (Table 4-2) are used by the SONIC during normal operation and are not intended to be accessed by the user.

**National Factory Test Registers:** These registers (Table 4-3) are for National factory use only and should never be accessed by the user. Accessing these registers during normal operation can cause improper functioning of the SONIC.

#### 4.1 THE CAM UNIT

The CAM unit memory cells are indirectly accessed by programming the CAM descriptor area in system memory and issuing the LCAM command (setting the LCAM bit in the Control register). The CAM cells do not occupy address locations in register space and, thus, are not accessible through the RA5-RA0 address pins. The CAM control registers, however, are part of the user register set and must be initialized before issuing the LCAM command (see Section 4.3.10).

The Content Addressable Memory (CAM) consists of sixteen 48-bit entries for complete address filtering (Figure 4-1) of network packets. Each entry corresponds to a 48-bit destination address that is user programmable and can contain any combination of Multicast or Physical addresses. Each entry is partitioned into three 16-bit CAM cells accessible through CAM Address Ports (CAP 2, CAP 1 and CAP 0) with CAP0 corresponding to the least significant 16 bits of the Destination Address and CAP2 corresponding to the most significant bits. The CAM is accessed in a two step process. First, the CAM Entry Pointer is loaded to point to one of the 16 entries. Then, each of the CAM Address Ports is accessed to select the CAM cell. The 16 user programmable CAM entries can be masked out with the CAM Enable register (see Section 4.3.10).

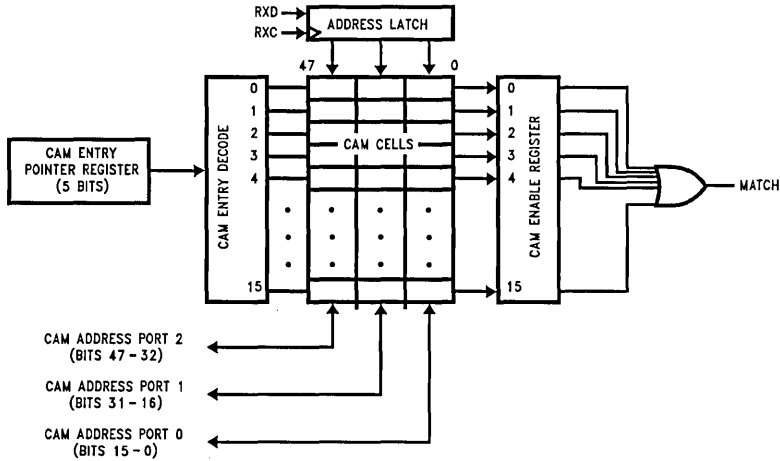
**Note:** It is not necessary to program a broadcast address into the CAM when it is desired to accept broadcast packets. Instead, to accept broadcast packets, set the BRD bit in the Receive Control register. If the BRD bit has been set, the CAM is still active. This means that it is possible to accept broadcast packets at the same time as accepting packets that match physical addresses in the CAM.

##### 4.1.1 The Load CAM Command

Because the SONIC uses the CAM for a relatively long period of time during reception, it can only be written to via the CAM Descriptor Area (CDA) and is only readable when the



## 4.0 SONIC Registers (Continued)



TL/F/10492-21

**FIGURE 4-1. CAM Organization**

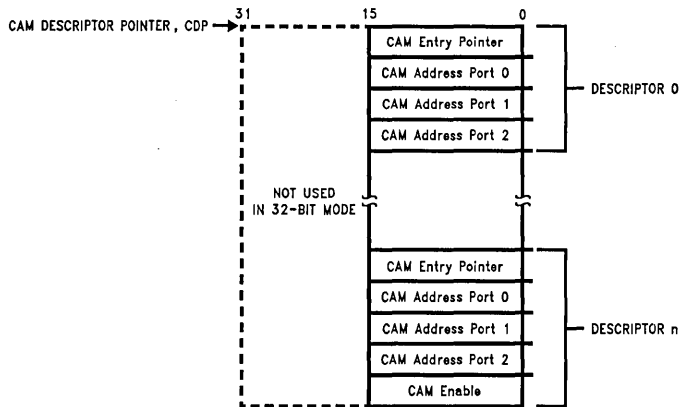
SONIC is in software reset. The CDA resides in the same 64k byte block of memory as the Receive Resource Area (RRA) and contains descriptors for loading the CAM registers. These descriptors are contiguous and each descriptor consists of four 16-bit fields (*Figure 4-2*). In 32-bit mode the upper word,  $D<31:16>$ , is not used. The first field contains the value to be loaded into the CAM Entry Pointer and the remaining fields are for the three CAM Address Ports (see Section 4.3.10). In addition, there is one more field after the last descriptor containing the mask for the CAM Enable register. Each of the CAM descriptors are addressed by the CAM Descriptor Pointer (CDP) register.

After the system has initialized the CDA, it can issue the Load CAM command to program the SONIC to read the CDA and load the CAM. The procedure for issuing the Load CAM command is as follows.

1. Initialize the Upper Receive Resource Address (URRA) register. Note that the CAM Descriptor Area must reside within the same 64k Page as the Receive Resource Area. (See Section 4.3.9).

2. Initialize the CDA as described above.
3. Initialize the CAM Descriptor Count with the number of CAM descriptors. Note, only the lower 5 bits are used in this register. The other bits are don't cares. (See Section 4.3.10).
4. Initialize the CAM Descriptor Pointer to locate the first descriptor in the CDA. This register must be reloaded each time a new Load CAM command is issued.
5. Issue the Load CAM command (LCAM) in the Command register. (See Section 4.3.1).

If a transmission or reception is in progress, the CAM DMA function will not occur until these operations are complete. When the SONIC completes the Load CAM command, the CDP register points to the next location after the CAM enable field and the CDC equals zero. The SONIC resets the LCAM bit in the Command register and sets the Load CAM Done (LCD) bit in the ISR.



TL/F/10492-22

**FIGURE 4-2. CAM Descriptor Area Format**

## 4.0 SONIC Registers (Continued)

RA <5:0>		15	0				
Status and Control Registers	0h Command Register	Status and C Control Fields					
	1 Data Configuration Register	Control Fields					
	2 Receive Control Register	Status and Control Fields					
	3 Transmit Control Register	Status and Control Fields					
	4 Interrupt Mask Register	Mask Fields					
	5 Interrupt Status Register	Status Fields					
Transmit Registers	3F Data Configuration Register 2	Control Fields					
	6 Upper Transmit Descriptor Address Register	Upper 16-bit Address Base					
Receive Registers	7 Current Transmit Descriptor Address Register	Lower 16-bit Address Offset					
	0D Upper Receive Descriptor Address Register	Upper 16-bit Address Base					
	0E Current Receive Descriptor Address Register	Lower 16-bit Address Offset					
	14 Upper Receive Resource Address Register	Upper 16-bit Address Base					
	15 Resource Start Address Register	Lower 16-bit Address Offset					
	16 Resource End Address Register	Lower 16-bit Address Offset					
	17 Resource Read Register	Lower 16-bit Address Offset					
	18 Resource Write Register	Lower 16-bit Address Offset					
	2B Receive Sequence Counter	Count Value	8   7	Count Value			
	CAM Registers	21 CAM Entry Pointer	<table border="1" style="width: 100%; text-align: right;"> <tr> <td style="width: 80%;"></td> <td style="width: 20%;">4</td> </tr> <tr> <td>Pointer</td> <td></td> </tr> </table>			4	Pointer
		4					
Pointer							
22 CAM Address Port 2		Most Significant 16 bits of CAM Entry					
23 CAM Address Port 1		Middle 16 bits of CAM Entry					
24 CAM Address Port 0		Least Significant 16 bits of CAM Entry					
25 CAM Enable Register	Mask Fields						
26 CAM Descriptor Pointer	Lower 16-bit Address Offset						
Tally Counters	27 CAM Descriptor Count	<table border="1" style="width: 100%; text-align: right;"> <tr> <td style="width: 80%;"></td> <td style="width: 20%;">5</td> </tr> <tr> <td>Count Value</td> <td></td> </tr> </table>			5	Count Value	
		5					
	Count Value						
2C CRC Error Tally Counter	Count Value						
2D Frame Alignment Error Tally	Count Value						
Watchdog Timer	2E Missed Packet Tally	Count Value					
	29 Watchdog Timer 0	Lower 16-bit Count Value					
	2A Watchdog Timer 1	Upper 16-bit Count Value					
	28 Silicon Revision Register	Chip Revision Number					

**FIGURE 4-3. Register Programming Model**

## 4.0 SONIC Registers (Continued)

### 4.2 STATUS/CONTROL REGISTERS

This set of registers is used to convey status/control information to/from the host system and to control the operation of the SONIC. These registers are used for loading commands generated from the system, indicating transmit and receive status, buffering data to/from memory, and provid-

ing interrupt control. The registers are selected by asserting chip select to the SONIC and providing the necessary address on register address pins RA5-RA0. Tables 4-1, 4-2, and 4-3 show the locations of all SONIC registers and where information on the registers can be found in the data sheet.

TABLE 4-1. User Registers

RA5-RA0	Access	Register	Symbol	Description (section)
<b>COMMAND AND STATUS REGISTERS</b>				
00h	R/W	Command	CR	4.3.1
01 (Note 3)	R/W	Data Configuration	DCR	4.3.2
02	R/W	Receive Control	RCR	4.3.3
03	R/W	Transmit Control	TCR	4.3.4
04	R/W	Interrupt Mask	IMR	4.3.5
05	R/W	Interrupt Status	ISR	4.3.6
3F (Note 3)	R/W	Data Configuration 2	DCR2	4.3.7
<b>TRANSMIT REGISTERS</b>				
06	R/W	Upper Transmit Descriptor Address	UTDA	4.3.8, 3.4.4.1
07	R/W	Current Transmit Descriptor Address	CTDA	4.3.8, 3.5.3
<b>RECEIVE REGISTERS</b>				
0D	R/W	Upper Receive Descriptor Address	URDA	4.3.9, 3.4.4.1
0E	R/W	Current Receive Descriptor Address	CRDA	4.3.9, 3.4.4.3
13	R/W	End of Buffer Word Count	EOBC	4.3.9, 3.4.2
14	R/W	Upper Receive Resource Address	URRA	4.3.9, 3.4.4.1
15	R/W	Resource Start Address	RSA	4.3.9, 3.4.1
16	R/W	Resource End Address	REA	4.3.9, 3.4.1
17	R/W	Resource Read Pointer	RRP	4.3.9, 3.4.1
18	R/W	Resource Write Pointer	RWP	4.3.9, 3.4.1
2B	R/W	Receive Sequence Counter	RSC	4.3.9, 3.4.3.2
<b>CAM REGISTERS</b>				
21	R/W	CAM Entry Pointer	CEP	4.1, 4.3.10
22 (Note 1)	R	CAM Address Port 2	CAP2	4.1, 4.3.10
23 (Note 1)	R	CAM Address Port 1	CAP1	4.1, 4.3.10
24 (Note 1)	R	CAM Address Port 0	CAP0	4.1, 4.3.10
25 (Note 2)	R/W	CAM Enable	CE	4.1, 4.3.10
26	R/W	CAM Descriptor Pointer	CDP	4.1, 4.3.10
27	R/W	CAM Descriptor Count	CDC	4.1, 4.3.10
<b>TALLY COUNTERS</b>				
2C (Note 4)	R/W	CRC Error Tally	CRCT	4.3.11
2D (Note 4)	R/W	FAE Tally	FAET	4.3.11
2E (Note 4)	R/W	Missed Packet Tally	MPT	4.3.11

## 4.0 SONIC Registers (Continued)

**TABLE 4-1. User Registers (Continued)**

RA5-RA0	Access	Register	Symbol	Description (section)
<b>WATCHDOG COUNTERS</b>				
29	R/W	Watchdog Timer 0	WT0	4.3.12
2A	R/W	Watchdog Timer 1	WT1	4.3.12
<b>SILICON REVISION</b>				
28	R	Silicon Revision	SR	4.3.13
<p><b>Note 1:</b> These registers can only be read when the SONIC is in reset mode (RST bit in the CR is set). The SONIC gives invalid data when these registers are read in non-reset mode.</p> <p><b>Note 2:</b> This register can only be written to when the SONIC is in reset mode. This register is normally only loaded by the Load CAM command.</p> <p><b>Note 3:</b> The Data Configuration registers, DCR and DCR2, can only be written to when the SONIC is in reset mode (RST bit in CR is set). Writing to these registers while not in reset mode does not alter the registers.</p> <p><b>Note 4:</b> The data written to these registers is inverted before being latched. That is, if a value of FFFFh is written, these registers will contain and read back the value of 0000h. Data is not inverted during a read operation.</p>				

**TABLE 4-2. Internal Use Registers (Users should not write to these registers)**

(RA5-RA0)	Access	Register	Symbol	Description (section)
<b>TRANSMIT REGISTERS</b>				
08 (Note 1)	R/W	Transmit Packet Size	TPS	3.5
09	R/W	Transmit Fragment Count	TFC	3.5
0A	R/W	Transmit Start Address 0	TSA0	3.5
0B	R/W	Transmit Start Address 1	TSA1	3.5
0C (Note 2)	R/W	Transmit Fragment Size	TFS	3.5
20	R/W	Temporary Transmit Descriptor Address	TTDA	3.5.4
2F	R	Maximum Deferral Timer	MDT	4.3.4
<b>RECEIVE REGISTERS</b>				
0F	R/W	Current Receive Buffer Address 0	CRBA0	3.4.2, 3.4.4.2
10	R/W	Current Receive Buffer Address 1	CRBA1	3.4.2, 3.4.4.2
11	R/W	Remaining Buffer Word Count 0	RBWC0	3.4.2, 3.4.4.2
12	R/W	Remaining Buffer Word Count 1	RBWC1	3.4.2, 3.4.4.2
19	R/W	Temporary Receive Buffer Address 0	TRBA0	3.4.6.2
1A	R/W	Temporary Receive Buffer Address 1	TRBA1	3.4.6.2
1B	R/W	Temporary Buffer Word Count 0	TBWC0	3.4.6.2
1C	R/W	Temporary Buffer Word Count 1	TBWC1	3.4.6.2
1F	R/W	Last Link Field Address	LLFA	none
<b>ADDRESS GENERATORS</b>				
1D	R/W	Address Generator 0	ADDR0	none
1E	R/W	Address Generator 1	ADDR1	none

**Note 1:** The data that is read from these registers is the inversion of what has been written to them.

**Note 2:** The value that is written to this register is shifted once in 16-bit mode and shifted twice in 32-bit mode.

**TABLE 4-3. Internal Use Registers (Users should not access these registers)**

(RA5-RA0)	Access	Register	Symbol	Description (section)
30 • 3E	R/W	These registers are for factory use only. Users must not address these registers as improper SONIC operation can occur.	none	none

## 4.0 SONIC Registers (Continued)

### 4.3 REGISTER DESCRIPTION

#### 4.3.1 Command Register

(RA <5:0> = 0h)

This register (Figure 4-4) is used for issuing commands to the SONIC. These commands are issued by setting the corresponding bits for the function. For all bits, except for the RST bit, the SONIC resets the bit after the command is completed. With the exception of RST, writing a "0" to any bit has no effect. Before any commands can be issued, the RST bit must first be reset to "0". This means that, if the RST bit is set, two writes to the Command Register are required to issue a command to the SONIC; one to clear the RST bit, and one to issue the command.

This register also controls the general purpose 32-bit Watchdog Timer. After the Watchdog Timer register has been loaded, it begins to decrement once the ST bit has been set to "1". An interrupt is issued when the count reaches zero if the Timer Complete interrupt is enabled in the IMR.

During hardware reset, bits 7, 4, and 2 are set to a "1"; all others are cleared. During software reset bits 9, 8, 1, and 0 are cleared and bits 7 and 2 are set to a "1"; all others are unaffected.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	LCAM	RRRA	RST	0	ST	STP	RXEN	RXDIS	TXP	HTX
						r/w	r/w	r/w		r/w	r/w	r/w	r/w	r/w	r/w

r/w = read/write

**FIGURE 4-4. Command Register**

Field	Meaning
LCAM	LOAD CAM
RRRA	READ RRA
RST	SOFTWARE RESET
ST	START TIMER
STP	STOP TIMER
RXEN	RECEIVER ENABLE
RXDIS	RECEIVER DISABLE
TXP	TRANSMIT PACKET(S)
HTX	HALT TRANSMISSION

Bit	Description
15–10	Must be 0
9	<p><b>LCAM: LOAD CAM</b></p> <p>Setting this bit causes the SONIC to load the CAM with the descriptor that is pointed to by the CAM Descriptor Pointer register.</p> <p><b>Note:</b> This bit must not be set during transmission (TXP is set). The SONIC will lock up if both bits are set simultaneously.</p>
8	<p><b>RRRA: READ RRA</b></p> <p>Setting this bit causes the SONIC to read the next RRA descriptor pointed to by the Resource Read Pointer (RRP) register. Generally this bit is only set during initialization. Setting this bit during normal operation can cause improper receive operation.</p>
7	<p><b>RST: SOFTWARE RESET</b></p> <p>Setting this bit resets all internal state machines. The CRC generator is disabled and the Tally counters are halted, but not cleared. The SONIC becomes operational when this bit is reset to "0". A hardware reset sets this bit to a "1". It must be reset to "0" before the SONIC becomes operational.</p>
6	Must be 0.
5	<p><b>ST: START TIMER</b></p> <p>Setting this bit enables the general-purpose watchdog timer to begin counting or to resume counting after it has been halted. This bit is reset when the timer is halted (i.e., STP is set). Setting this bit resets STP.</p>
4	<p><b>STP: STOP TIMER</b></p> <p>Setting this bit halts the general-purpose watchdog timer and resets the ST bit. The timer resumes when the ST bit is set. This bit powers up as a "1". <b>Note:</b> Simultaneously setting bits ST and STP stops the timer.</p>

## 4.0 SONIC Registers (Continued)

### 4.3 REGISTER DESCRIPTION (Continued)

#### 4.3.1 Command Register (Continued)

Bit	Description
3	<p><b>RXEN: RECEIVER ENABLE</b></p> <p>Setting this bit enables the receive buffer management engine to begin buffering data to memory. Setting this bit resets the RXDIS bit. Note: If this bit is set while the MAC unit is currently receiving a packet, both RXEN and RXDIS are set until the network goes inactive (i.e., the SONIC will not start buffering in the middle of a packet being received).</p>
2	<p><b>RXDIS: RECEIVER DISABLE</b></p> <p>Setting this bit disables the receiver from buffering data to memory or the Receive FIFO. If this bit is set during the reception of a packet, the receiver is disabled only after the packet is processed. The RXEN bit is reset when the receiver is disabled. Tally counters remain active regardless of the state of this bit.</p> <p><b>Note:</b> If this bit is set while the SONIC is currently receiving a packet, both RXEN and RXDIS are set until the packet is fully received. When both RXEN and RXDIS are set, RXDIS could be cleared by writing zero to it.</p>
1	<p><b>TXP: TRANSMIT PACKET(S)</b></p> <p>Setting this bit causes the SONIC to transmit packets which have been set up in the Transmit Descriptor Area (TDA). The SONIC loads its appropriate registers from the TDA, then begins transmission. The SONIC clears this bit after any of the following conditions have occurred: (1) transmission had completed (i.e., after the SONIC has detected EOL = 1), (2) the Halt Transmission command (HTX) has taken effect, or (3) a transmit abort condition has occurred. This condition occurs when any of the following bits in the TCR have been set: EXC, EXD, FU, or BCM. This bit must not be set if a Load CAM operation is in progress (LCAM is set). The SONIC will lock up if both bits are set simultaneously.</p>
0	<p><b>HTX: HALT TRANSMISSION</b></p> <p>Setting this bit halts the transmit command after the current transmission has completed. TXP is reset after transmission has halted. The Current Transmit Descriptor Address (CTDA) register points to the last descriptor transmitted. The SONIC samples this bit after writing to the TXpkt.status field.</p>

## 4.0 SONIC Registers (Continued)

### 4.3.2 Data Configuration Register

(RA<5:0> = 1h)

This register (Figure 4-5) establishes the bus cycle options for reading/writing data to/from 16- or 32-bit memory systems. During a hardware reset, bits 15 and 13 are cleared; all other bits are unaffected. (Because of this, the first thing the driver software does to the SONIC should be to set up this register.) All bits are unaffected by a software reset. This register must only be accessed when the SONIC is in reset mode (i.e., the RST bit is set in the Command register).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXBUS	0	LBR	PO1	PO0	SBUS	USR1	USR0	WC1	WC0	DW	BMS	RFT1	RFT0	TFT1	TFT0
r/w		r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

r/w = read/write

FIGURE 4-5. Data Configuration Register

Field	Meaning
EXBUS	EXTENDED BUS MODE
LBR	LATCHED BUS RETRY
PO0,PO1	PROGRAMMABLE OUTPUTS
SBUS	SYNCHRONOUS BUS MODE
USR0, USR1	USER DEFINABLE PINS
WC0, WC1	WAIT STATE CONTROL
DW	DATA WIDTH SELECT
BMS	BLOCK MODE SELECT FOR DMA
RFT0, RFT1	RECEIVE FIFO THRESHOLD
TFT0, TFT1	TRANSMIT FIFO THRESHOLD

Bits	Description
15	<p><b>EXBUS: EXTENDED BUS MODE</b></p> <p>Setting this bit enables the Extended Bus mode which enables the following:</p> <ol style="list-style-type: none"> <li>1)Extended Programmable Outputs, EXUSR &lt;3:0&gt;: This changes the TXD, LBK, RXC and RXD pins from the external ENDEC interface into four programmable user outputs, EXUSR &lt;3:0&gt; respectively, which are similar to USR &lt;1:0&gt;. These outputs are programmed with bits 15-12 in the DCR2 (see Section 4.3.7). On hardware reset, these four pins will be TRI-STATE and will remain that way until the DCR is changed. If EXBUS is enabled, then these pins will remain TRI-STATE until the SONIC becomes a bus master, at which time they will be driven according to the DCR2. If EXBUS is disabled, then these four pins work normally as external ENDEC interface pins.</li> <li>2)Synchronous Termination, STERM: This changes the TXC pin from the External ENDEC interface into a synchronous memory termination input for compatibility with Motorola style processors. This input is only useful when Asynchronous Bus mode is selected (bit 10 below is set to "0") and BMODE = 1 (Motorola mode). On hardware reset, this pin will be TRI-STATE and will remain that way until the DCR is changed. If EXBUS is enabled, this pin will remain TRI-STATE until the SONIC becomes a bus master, at which time it will become the STERM input. If EXBUS is disabled, then this pin works normally as the TXC pin for the external ENDEC interface.</li> <li>3)Asynchronous Bus Retry: Causes <math>\overline{BRT}</math> to be clocked in asynchronously off the falling edge of bus clock. This only applies, however, when the SONIC is operating in asynchronous mode (bit 10 below is set to "0"). If EXBUS is not set, XTO (BRT) is sampled synchronously off the rising edge of bus clock. (See Section 5.4.6.)</li> </ol>
14	Must be 0.
13	<p><b>LBR: LATCHED BUS RETRY</b></p> <p>The LBR bit controls the mode of operation of the <math>\overline{BRT}</math> signal (see pin description). It allows the BUS Retry operation to be latched or unlatched.</p> <p>0:Unlatched mode: The assertion of <math>\overline{BRT}</math> forces the SONIC to finish the current DMA operation and get off the bus. The SONIC will retry the operation when <math>\overline{BRT}</math> is desasserted.</p> <p>1:Latched mode: The assertion of <math>\overline{BRT}</math> forces the SONIC to finish the current DMA operation as above, however, the SONIC will not retry until <math>\overline{BRT}</math> is deasserted, the BR bit in the ISR (see Section 4.3.6) has been reset, and <math>\overline{BRT}</math> is deasserted. Hence, the mode has been latched on until the BR bit is cleared.</p> <p><b>Note:</b> Unless LBR is set to a "1", <math>\overline{BRT}</math> must remain asserted at least until the SONIC has gone idle. See Section 5.4.6 and the timing for Bus Retry in section 7.0.</p>
12, 11	<p><b>PO1, PO0: PROGRAMMABLE OUTPUTS</b></p> <p>The PO1,PO0 bits individually control the USR1,0 pins respectively when SONIC is a bus master (HLDA or <math>\overline{BGACK}</math> is active). When PO1/PO0 are set to a 1 the USR1/USR0 pins are high during bus master operations and when these bits are set to a 0 the USR1/USR0 pins are low during bus master operations.</p>

1

## 4.0 SONIC Registers (Continued)

### 4.3.2 Data Configuration Register (Continued)

Bits	Description															
10	<p><b>SBUS: SYNCHRONOUS BUS MODE</b></p> <p>The SBUS bit is used to select the mode of system bus operation when SONIC is a bus master. This bit selects the internal ready line to be either a synchronous or asynchronous input to SONIC during block transfer DMA operations.</p> <p>0: Asynchronous mode. <math>\overline{RDYi}</math> (BMODE = 0) or <math>\overline{DSACK0,1}</math> (BMODE = 1) are respectively internally synchronized at the falling edge of the bus clock (T2 of the DMA cycle). No setup or hold times need to be met with respect to this edge to guarantee proper bus operation. The minimum memory cycle time is 3 bus clocks.</p> <p>1: Synchronous mode. <math>\overline{RDYi}</math> (BMODE = 0) and <math>\overline{DSACK0,1}</math> (BMODE = 1) must respectively meet the setup and hold times with respect to the rising edge of T1 or T2 to guarantee proper bus operation.</p>															
9, 8	<p><b>USR1,0: USER DEFINABLE PINS</b></p> <p>The USR1,0 bits report the level of the USR1,0 signal pins, respectively, after a chip hardware reset. If the USR1,0 signal pins are at a logical 1 (tied to <math>V_{CC}</math>) during a hardware reset the USR1,0 bits are set to a 1. If the USR1,0 pins are at a logical 0 (tied to ground) during a hardware reset the USR1,0 bits are set to a 0. These bits are latched on the rising edge of RST. Once set they remain set/reset until the next hardware reset.</p>															
7, 6	<p><b>WC1,0: WAIT STATE CONTROL</b></p> <p>These encoded bits determine the number of additional bus cycles (T2 states) that are added during each DMA cycle.</p> <table style="margin-left: 20px;"> <thead> <tr> <th style="text-align: center;">WC1</th> <th style="text-align: center;">WC0</th> <th style="text-align: center;">Bus Cycles Added</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">2</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">3</td> </tr> </tbody> </table>	WC1	WC0	Bus Cycles Added	0	0	0	0	1	1	1	0	2	1	1	3
WC1	WC0	Bus Cycles Added														
0	0	0														
0	1	1														
1	0	2														
1	1	3														
5	<p><b>DW: DATA WIDTH SELECT</b></p> <p>These bits select the data path width for DMA operations.</p> <table style="margin-left: 20px;"> <thead> <tr> <th style="text-align: center;">DW</th> <th style="text-align: center;">Data Width</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">16-bit</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">32-bit</td> </tr> </tbody> </table>	DW	Data Width	0	16-bit	1	32-bit									
DW	Data Width															
0	16-bit															
1	32-bit															
4	<p><b>BMS: BLOCK MODE SELECT FOR DMA</b></p> <p>Determines how data is emptied or filled into the Receive or Transmit FIFO.</p> <p>0: Empty/fill mode: All DMA transfers continue until either the Receive FIFO has emptied or the Transmit FIFO has filled completely.</p> <p>1: Block mode: All DMA transfers continue until the programmed number of bytes (RFT0, RFT1 during reception or TF0, TF1 during transmission) have been transferred. (See note for TFT0, TFT1.)</p>															
3, 2	<p><b>RFT1,RFT0: RECEIVE FIFO THRESHOLD</b></p> <p>These encoded bits determine the number of words (or long words) that are written into the receive FIFO from the MAC unit before a receive DMA request occurs. (See Section 1.4.)</p> <table style="margin-left: 20px;"> <thead> <tr> <th style="text-align: center;">LB1</th> <th style="text-align: center;">LB0</th> <th style="text-align: center;">Function</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">2 words or 1 long word (4 bytes)</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">4 words or 2 long words (8 bytes)</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">8 words or 4 long words (16 bytes)</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">12 words or 6 long words (24 bytes)</td> </tr> </tbody> </table> <p><b>Note:</b> In block mode (BMS bit = 1), the receive FIFO threshold sets the number of words (or long words) written to memory during a receive DMA block cycle.</p>	LB1	LB0	Function	0	0	2 words or 1 long word (4 bytes)	0	1	4 words or 2 long words (8 bytes)	1	0	8 words or 4 long words (16 bytes)	1	1	12 words or 6 long words (24 bytes)
LB1	LB0	Function														
0	0	2 words or 1 long word (4 bytes)														
0	1	4 words or 2 long words (8 bytes)														
1	0	8 words or 4 long words (16 bytes)														
1	1	12 words or 6 long words (24 bytes)														
1, 0	<p><b>TFT1,TFT0: TRANSMIT FIFO THRESHOLD</b></p> <p>These encoded bits determine the minimum number of words (or long words) the DMA section maintains in the transmit FIFO. A bus request occurs when the number of words drops below the transmit FIFO threshold. (See Section 1.4.)</p> <table style="margin-left: 20px;"> <thead> <tr> <th style="text-align: center;">LB1</th> <th style="text-align: center;">LB0</th> <th style="text-align: center;">Function</th> </tr> </thead> <tbody> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">0</td> <td style="text-align: center;">4 words or 2 long words (8 bytes)</td> </tr> <tr> <td style="text-align: center;">0</td> <td style="text-align: center;">1</td> <td style="text-align: center;">8 words or 4 long words (16 bytes)</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">0</td> <td style="text-align: center;">12 words or 6 long words (24 bytes)</td> </tr> <tr> <td style="text-align: center;">1</td> <td style="text-align: center;">1</td> <td style="text-align: center;">14 words or 7 long words (28 bytes)</td> </tr> </tbody> </table> <p><b>Note:</b> In block mode (BMS = 1), the number of bytes the SONIC reads in a single DMA burst equals the transmit FIFO threshold value. If the number of words or long words needed to fill the FIFO is less than the threshold value, then only the number of reads required to fill the FIFO in a single DMA burst will be made. Typically, with the FIFO threshold value set to 12 or 14 words, the number of memory reads needed is less than the FIFO threshold value.</p>	LB1	LB0	Function	0	0	4 words or 2 long words (8 bytes)	0	1	8 words or 4 long words (16 bytes)	1	0	12 words or 6 long words (24 bytes)	1	1	14 words or 7 long words (28 bytes)
LB1	LB0	Function														
0	0	4 words or 2 long words (8 bytes)														
0	1	8 words or 4 long words (16 bytes)														
1	0	12 words or 6 long words (24 bytes)														
1	1	14 words or 7 long words (28 bytes)														



## 4.0 SONIC Registers (Continued)

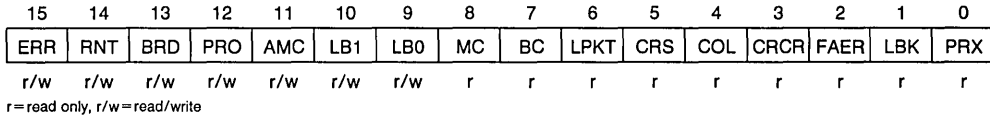
### 4.3.3 Receive Control Register

(RA <5:0> = 2h)

This register is used to filter incoming packets and provide status information of accepted packets (Figure 4-6). Setting any of bits 15–11 to a “1” enables the corresponding receive filter. If none of these bits are set, only packets which match the CAM Address registers are accepted. Bits 10 and 9 control the loopback operations.

After reception, bits 8–0 indicate status information about the accepted packet and are set to “1” when the corresponding condition is true. If the packet is accepted, all bits in the RCR are written into the RXpkt.status field. Bits 8–6 and 3–0 are cleared at the reception of the next packet.

This register is unaffected by a software reset.



**FIGURE 4-6. Receive Control Register**

Field	Meaning
ERR	ACCEPT PACKET WITH ERRORS
RNT	ACCEPT RUNT PACKETS
BRD	ACCEPT BROADCAST PACKETS
PRO	PHYSICAL PROMISCUOUS PACKETS
AMC	ACCEPT ALL MULTICAST PACKETS
LB0, LB1	LOOPBACK CONTROL
MC	MULTICAST PACKET RECEIVED
BC	BROADCAST PACKET RECEIVED
LPKT	LAST PACKET IN RBA
CRS	CARRIER SENSE ACTIVITY
COL	COLLISION ACTIVITY
CRCR	CRC ERROR
FAER	FRAME ALIGNMENT ERROR
LBK	LOOPBACK PACKET RECEIVED
PRX	PACKET RECEIVED OK

Bit	Description
15	<p><b>ERR: ACCEPT PACKET WITH CRC ERRORS OR COLLISIONS</b>                      0: Reject all packets with CRC errors or when a collision occurs.                      1: Accept packets with CRC errors and ignore collisions.</p>
14	<p><b>RNT: ACCEPT RUNT PACKETS</b>                      0: Normal address match mode.                      1: Accept runt packets (packets less than 64 bytes in length).                      Note: A hardware reset clears this bit.</p>
13	<p><b>BRD: ACCEPT BROADCAST PACKETS</b>                      0: Normal address match mode.                      1: Accept broadcast packets (packets with addresses that match the CAM are also accepted).                      Note: This bit is cleared upon hardware reset.</p>
12	<p><b>PRO: PHYSICAL PROMISCUOUS MODE</b>                      Enable all Physical Address packets to be accepted.                      0: normal address match mode.                      1: promiscuous mode.</p>
11	<p><b>AMC: ACCEPT ALL MULTICAST PACKETS</b>                      0: normal address match mode.                      1: enables all multicast packets to be accepted. Broadcast packets are also accepted regardless of the BRD bit. (Broadcast packets are a subset of multicast packets.)</p>

## 4.0 SONIC Registers (Continued)

### 4.3.3 Receive Control Register (Continued)

Bits	Description															
10, 9	<p><b>LB1, LB0: LOOPBACK CONTROL</b></p> <p>These encoded bits control loopback operations for MAC loopback, ENDEC loopback and Transceiver loopback. For proper loopback operation, the CAM Address registers and Receive Control register must be initialized to accept the Destination address of the loopback packet (see Section 1.7).</p> <p>Note: A hardware reset clears these bits.</p> <table border="1"> <thead> <tr> <th>LB1</th> <th>LB0</th> <th>Function</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>no loopback, normal operation</td> </tr> <tr> <td>0</td> <td>1</td> <td>MAC loopback</td> </tr> <tr> <td>1</td> <td>0</td> <td>ENDEC loopback</td> </tr> <tr> <td>1</td> <td>1</td> <td>Transceiver loopback</td> </tr> </tbody> </table>	LB1	LB0	Function	0	0	no loopback, normal operation	0	1	MAC loopback	1	0	ENDEC loopback	1	1	Transceiver loopback
LB1	LB0	Function														
0	0	no loopback, normal operation														
0	1	MAC loopback														
1	0	ENDEC loopback														
1	1	Transceiver loopback														
8	<p><b>MC: MULTICAST PACKET RECEIVED</b></p> <p>This bit is set when a packet is received with a Multicast Address.</p>															
7	<p><b>BC: BROADCAST PACKET RECEIVED</b></p> <p>This bit is set when a packet is received with a Broadcast Address.</p>															
6	<p><b>LPKT: LAST PACKET IN RBA</b></p> <p>This bit is set when the last packet is buffered into a Receive Buffer Area (RBA). The SONIC detects this condition when its Remaining Buffer Word Count (RBWC0,1) register is less than or equal to the End Of Buffer Count (EOBC) register. (See Section 3.4.2.)</p>															
5	<p><b>CRS: CARRIER SENSE ACTIVITY</b></p> <p>Set when CRS is active. Indicates the presence of network activity.</p>															
4	<p><b>COL: COLLISION ACTIVITY</b></p> <p>Indicates that the packet received had a collision occur during reception.</p>															
3	<p><b>CRCR: CRC ERROR</b></p> <p>Indicates the packet contains a CRC error. If the packet also contains a Frame Alignment error, FAER will be set instead (see bit 2, below). The rev C SONIC has the potential to report CRC error'd frames as FAE. This bit is also not set during CRC error'd RUNT packets.</p>															
2	<p><b>FAER: FRAME ALIGNMENT ERROR</b></p> <p>Indicates that the incoming packet was not correctly framed on an 8-bit boundary. Note: if no CRC errors have occurred, this bit is not set (i.e., this bit is only set when both a frame alignment and CRC error occurs).</p>															
1	<p><b>LBK: LOOPBACK PACKET RECEIVED</b></p> <p>Indicates that the SONIC has successfully received a loopback packet.</p>															
0	<p><b>PRX: PACKET RECEIVED OK</b></p> <p>Indicates that a packet has been received without CRC, frame alignment, length (runt packet) errors or collisions.</p>															

## 4.0 SONIC Registers (Continued)

### 4.3.4 Transmit Control Register

(RA<5:0> = 3h)

This register is used to program the SONIC's transmit actions and provide status information after a packet has been transmitted (Figure 4-7). At the beginning of transmission, bits 15, 14, 13 and 12 from the TXpkt.config field are loaded into the TCR to configure the various transmit modes (see Section 3.5.1.1). When the transmission ends, bits 10–0 indicate status information and are set to a "1" when the corresponding condition is true. These bits, along with the number of collisions information, are written into the TXpkt.status field at the end of transmission (see Section 3.5.1.2). Bits 9 and 5 are cleared after the TXpkt.status field has been written. Bits 10, 7, 6, and 1 are cleared at the commencement of the next transmission while bit 8 is set at this time.

A hardware reset sets bits 8 and 0 to a "1" and bit 1 to a 0. This register is unaffected by a software reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PINT	POWC	CRCI	EXDIS	0	EXD	DEF	NCRS	CRSL	EXC	OWC	0	PMB	FU	BCM	PTX
r/w	r/w	r/w	r/w	r	r	r	r	r	r	r	r	r	r	r	r

r = read only, r/w = read/write

**FIGURE 4-7. Transmit Control Register**

Field	Meaning
PINT	PROGRAMMABLE INTERRUPT
POWC	PROGRAMMED OUT OF WINDOW COLLISION TIMER
CRCI	CRC INHIBIT
EXDIS	DISABLE EXCESSIVE DEFERRAL TIMER
EXD	EXCESSIVE DEFERRAL
DEF	DEFERRED TRANSMISSION
NCRS	NO CRS
CRSL	CRS LOST
EXC	EXCESSIVE COLLISIONS
OWC	OUT OF WINDOW COLLISION
PMB	PACKET MONITORED BAD
FU	FIFO UNDERRUN
BCM	BYTE COUNT MISMATCH
PTX	PACKET TRANSMITTED OK

Bit	Description
15	<p><b>PINT: PROGRAMMABLE INTERRUPT</b></p> <p>This bit allows transmit interrupts to be generated under software control. The SONIC will issue an interrupt (PINT in the Interrupt Status Register) immediately after reading a TDA and detecting that PINT is set in the TXpkt.config field.</p> <p><b>Note:</b> In order for PINT to operate properly, it must be set and reset in the TXpkt.config field by alternating TDAs. This is necessary because after PINT has been issued in the ISR, PINT in the Transmit Control Register must be cleared before it is set again in order to have the interrupt issued for another packet. The only effective way to do this is to set PINT to a 1 no more often than every other packet.</p>
14	<p><b>POWC: PROGRAM "OUT OF WINDOW COLLISION" TIMER</b></p> <p>This bit programs when the out of window collision timer begins.</p> <p>0: timer begins after the Start of Frame Delimiter (SFD).</p> <p>1: timer begins after the first bit of preamble.</p>
13	<p><b>CRCI: CRC INHIBIT</b></p> <p>0: transmit packet with 4-byte FCS field.</p> <p>1: transmit packet without 4-byte FCS field.</p>
12	<p><b>EXDIS: DISABLE EXCESSIVE DEFERRAL TIMER:</b></p> <p>0: excessive deferral timer enabled.</p> <p>1: excessive deferral timer disabled.</p>
11	Must be 0.
10	<p><b>EXD: EXCESSIVE DEFERRAL</b></p> <p>Indicates that the SONIC has been deferring for 3.2 ms. The transmission is aborted if the excessive deferral timer is enabled (i.e. EXDIS is reset). This bit can only be set if the excessive deferral timer is enabled.</p>

## 4.0 SONIC Registers (Continued)

### 4.3.4 Transmit Control Register (Continued)

Bit	Description
9	<p><b>DEF: DEFERRED TRANSMISSION</b></p> <p>Indicates that the SONIC has deferred its transmission during the first attempt. If subsequent collisions occur, this bit is reset. This bit is cleared after the TXpkt.status field is written in the TDA.</p>
8	<p><b>NCRS: NO CRS</b></p> <p>Indicates that Carrier Sense (CRS) was not present during transmission. CRS is monitored from the beginning of the Start of Frame Delimiter to the last byte transmitted. The transmission will not be aborted. This bit is set at the start of preamble and is reset if CRS is detected. Hence, if CRS is never detected throughout the entire transmission of the packet, this bit will remain set.</p> <p><b>Note:</b> NCRS will remain set in MAC loopback as long as there is no activity on the RX<math>\pm</math>.</p>
7	<p><b>CRSL: CRS LOST</b></p> <p>Indicates that CRS has gone low or has not been present during transmission. CRS is monitored from the beginning of the Start of Frame Delimiter to the last byte transmitted. The transmission will not be aborted.</p> <p><b>Note:</b> If CRS was never present, both NCRS and CRSL will be set simultaneously. Also, CRSL will always be set in MAC loopback.</p>
6	<p><b>EXC: EXCESSIVE COLLISIONS</b></p> <p>Indicates that 16 collisions have occurred. The transmission is aborted.</p>
5	<p><b>OWC: OUT OF WINDOW COLLISION</b></p> <p>Indicates that an illegal collision has occurred after 51.2 <math>\mu</math>s (one slot time) from either the first bit of preamble or from SFD depending upon the POWC bit. The transmission backs off as in a normal transmission. This bit is cleared after the TXpkt.status field is written in the TDA.</p>
4	Must be 0.
3	<p><b>PMB: PACKET MONITORED BAD</b></p> <p>This bit is set, if after the receive unit has monitored the transmitted packet, the CRC has been calculated as invalid as a result of a frame alignment error, or the Source Address does not match any of the CAM address registers.</p> <p><b>Note 1:</b> The SONIC's CRC checker is active during transmission.</p> <p><b>Note 2:</b> If CRC has been inhibited for transmissions (CRCI is set), this bit will always be low. This is true regardless of Frame Alignment or Source Address mismatch errors.</p> <p><b>Note 3:</b> If a Receive FIFO overrun has occurred, the transmitted packet is not monitored completely. Thus, if PMB bit is set along with the RFO bit in the ISR, then PMB has no meaning. The packet must be completely received before PMB has meaning.</p> <p><b>Note 4:</b> This bit is always zero in MAC, ENDEC, and Transceiver loopback modes.</p>
2	<p><b>FU: FIFO UNDERRUN</b></p> <p>Indicates that the SONIC has not been able to access the bus before the FIFO has emptied. This condition occurs from excessive bus latency and/or slow bus clock. The transmission is aborted. (See Section 1.4.2.)</p>
1	<p><b>BCM: BYTE COUNT MISMATCH</b></p> <p>This bit is set when the SONIC detects that the TXpkt.pkt_size field is not equal to the sum of the TXpkt.frag_size field(s). Transmission is aborted. This bit will also be set when Excessive Collisions (bit 6 of the transmit control register) occur during transmission.</p>
0	<p><b>PTX: PACKET TRANSMITTED OK</b></p> <p>Indicates that a packet has been transmitted without the following errors:</p> <ul style="list-style-type: none"> <li>—Excessive Collisions (EXC)</li> <li>—Excessive Deferral (EXD)</li> <li>—FIFO Underrun (FU)</li> <li>—Byte Count Mismatch (BCM)</li> </ul>

## 4.0 SONIC Registers (Continued)

### 4.3.5 Interrupt Mask Register

(RA <5:0> = 4h)

This register masks the interrupts that can be generated from the ISR (Figure 4-8). Writing a "1" to the bit enables the corresponding interrupt. During a hardware reset, all mask bits are cleared.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	BREN	HBLEN	LCDEN	PINTEN	PRXEN	PTXEN	TXEREN	TCEN	RDEEN	RBEEN	RBAEEN	CRCCEN	FAEEN	MPEN	RFOEN
	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

r/w = read/write

**FIGURE 4-8. Interrupt Mask Register**

Field	Meaning
BREN	BUS RETRY OCCURRED ENABLE
HBLEN	HEARTBEAT LOST ENABLE
LCDEN	LOAD CAM DONE INTERRUPT ENABLE
PINTEN	PROGRAMMABLE INTERRUPT ENABLE
PRXEN	PACKET RECEIVED ENABLE
PTXEN	PACKET TRANSMITTED OK ENABLE
TXEREN	TRANSMIT ERROR ENABLE
TCEN	TIMER COMPLETE ENABLE
RDEEN	RECEIVE DESCRIPTORS ENABLE
RBEEN	RECEIVE BUFFERS EXHAUSTED ENABLE
RBAEEN	RECEIVE BUFFER AREA EXCEEDED ENABLE
CRCCEN	CRC TALLY COUNTER WARNING ENABLE
FAEEN	FAE TALLY COUNTER WARNING ENABLE
MPEN	MP TALLY COUNTER WARNING ENABLE
RFOEN	RECEIVE FIFO OVERRUN ENABLE

Bit	Description
15	Must be 0.
14	<b>BREN: BUS RETRY OCCURRED enable:</b> 0: disable 1: enables interrupts when a Bus Retry operation is requested.
13	<b>HBLEN: HEARTBEAT LOST enable:</b> 0: disable 1: enables interrupts when a heartbeat lost condition occurs.
12	<b>LCDEN: LOAD CAM DONE INTERRUPT enable:</b> 0: disable 1: enables interrupts when the Load CAM command has finished.
11	<b>PINTEN: PROGRAMMABLE INTERRUPT enable:</b> 0: disable 1: enables programmable interrupts to occur when the PINT bit the TXpkt.config field is set to a "1".
10	<b>PRXEN: PACKET RECEIVED enable:</b> 0: disable 1: enables interrupts for packets accepted.
9	<b>PTXEN: PACKET TRANSMITTED OK enable:</b> 0: disable 1: enables interrupts for transmit completions.
8	<b>TXEREN: TRANSMIT ERROR enable:</b> 0: disable 1: enables interrupts for packets transmitted with error.

## 4.0 SONIC Registers (Continued)

### 4.3.5 Interrupt Mask Register (Continued)

Bit	Description
7	<b>TCEN: GENERAL PURPOSE TIMER COMPLETE enable:</b> 0: disable 1: enables interrupts when the general purpose timer has rolled over from 0000 0000h to FFFF FFFFh.
6	<b>RDEEN: RECEIVE DESCRIPTORS EXHAUSTED enable:</b> 0: disable 1: enables interrupts when all receive descriptors in the RDA have been exhausted.
5	<b>RBEEN: RECEIVE BUFFERS EXHAUSTED enable:</b> 0: disable 1: enables interrupts when all resource descriptors in the RRA have been exhausted.
4	<b>RBAAEN: RECEIVE BUFFER AREA EXCEEDED enable:</b> 0: disable 1: enables interrupts when the SONIC attempts to buffer data beyond the end of the Receive Buffer Area.
3	<b>CRCEEN: CRC TALLY COUNTER WARNING enable:</b> 0: disable 1: enables interrupts when the CRC tally counter has rolled over from FFFFh to 0000h.
2	<b>FAEEN: FRAME ALIGNMENT ERROR (FAE) TALLY COUNTER WARNING enable:</b> 0: disable 1: enables interrupts when the FAE tally counter rolled over from FFFFh to 0000h.
1	<b>MPEN: MISSED PACKET (MP) TALLY COUNTER WARNING enable:</b> 0: disable 1: enables interrupts when the MP tally counter has rolled over from FFFFh to 0000h.
0	<b>RFOEN: RECEIVE FIFO OVERRUN enable:</b> 0: disable 1: enables interrupts when the receive FIFO has overrun.

## 4.0 SONIC Registers (Continued)

### 4.3.6 Interrupt Status Register

(RA<5:0> = 5h)

This register (Figure 4-9) indicates the source of an interrupt when the INT pin goes active. Enabling the corresponding bits in the IMR allows bits in this register to produce an interrupt. When an interrupt is active, one or more bits in this register are set to a "1". A bit is cleared by writing "1" to it. Writing a "0" to any bit has no effect.

This register is cleared by a hardware reset and unaffected by a software reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	BR	HBL	LCD	PINT	PKTRX	TXDN	TXER	TC	RDE	RBE	RBAE	CRC	FAE	MP	RFO
	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

r/w = read/write

**FIGURE 4-9. Interrupt Status Register**

Field	Meaning
BR	BUS RETRY OCCURRED
HBL	CD HEARTBEAT LOST
LCD	LOAD CAM DONE
PINT	PROGRAMMABLE INTERRUPT
PKTRX	PACKET RECEIVED
TXDN	TRANSMISSION DONE
TXER	TRANSMIT ERROR
TC	TIMER COMPLETE
RDE	RECEIVE DISCRIPTORS EXHAUSTED
RBE	RECEIVE BUFFERS EXHAUSTED
RBAE	RECEIVE BUFFER AREA EXCEEDED
CRC	CRC TALLY COUNTER ROLLOVER
FAE	FRAME ALIGNMENT ERROR
MP	MISSED PACKET COUNTER ROLLOVER
RFO	RECEIVE FIFO OVERRUN

Bit	Description
15	Must be 0.
14	<b>BR: BUS RETRY OCCURRED</b> Indicates that a Bus Retry (BRT) operation has occurred. In Latched Bus Retry mode (LBR in the DCR), BR will only be set when the SONIC is a bus master. Before the SONIC will continue any DMA operations, BR must be cleared. In Unlatched mode, the BR bit should be cleared also, but the SONIC will not wait for BR to be cleared before requesting the bus again and continuing its DMA operations. (See Sections 4.3.2 and 5.4.6 for more information on Bus Retry.)
13	<b>HBL: CD HEARTBEAT LOST</b> If the transceiver fails to provide a collision pulse (heart beat) during the first 6.4 μs of the Interframe Gap after transmission, this bit is set.
12	<b>LCD: LOAD CAM DONE</b> Indicates that the Load CAM command has finished writing to all programmed locations in the CAM. (See Section 4.1.1.)
11	<b>PINT: PROGRAMMED INTERRUPT</b> Indicates that upon reading the TXpkt.config field, the SONIC has detected the PINT bit to be set. (See Section 4.3.4.)
10	<b>PKTRX: PACKET RECEIVED</b> Indicates that a packet has been received and been buffered to memory. This bit is set after the RXpkt.seq_no field is written to memory.
9	<b>TXDN: TRANSMISSION DONE</b> Indicates that either (1) there are no remaining packets to be transmitted in the Transmit Descriptor Area (i.e., the EOL bit has been detected as a "1"), (2) the Halt Transmit command has been given (HTX bit in CR is set to a "1"), or (3) a transmit abort condition has occurred. This condition occurs when any of following bits in the TCR are set: BCM, EXC, FU, or EXD. This bit is set after the TXpkt.status field has been written to.

## 4.0 SONIC Registers (Continued)

### 4.3.6 Interrupt Status Register (Continued)

Bit	Description
8	<p><b>TXER: TRANSMIT ERROR</b></p> <p>Indicates that a packet has been transmitted with at least one of the following errors.</p> <ul style="list-style-type: none"> <li>—Byte count mismatch (BCM)</li> <li>—Excessive collisions (EXC)</li> <li>—FIFO underrun (FU)</li> <li>—Excessive deferral (EXD)</li> </ul> <p>The TXpkt.status field reveals the cause of the error(s).</p>
7	<p><b>TC: GENERAL PURPOSE (Watchdog) TIMER COMPLETE</b></p> <p>Indicates that the timer has rolled over from 0000 0000h to FFFF FFFFh. (See Section 4.3.12.)</p>
6	<p><b>RDE: RECEIVE DESCRIPTORS EXHAUSTED</b></p> <p>Indicates that all receive packet descriptors in the RDA have been exhausted. This bit is set when the SONIC detects EOL = 1. (See Section 3.4.7.)</p>
5	<p><b>RBE: RECEIVE BUFFER EXHAUSTED</b></p> <p>Indicates that the SONIC has detected the Resource Read Pointer (RRP) is equal to the Resource Write Pointer (RWP). This bit is set after the last field is read from the resource area. (See Section 3.4.7.)</p> <p><b>Note 1:</b> This bit will be set as the SONIC finishes using the second to last receive buffer and reads the last RRA descriptor. This gives the system an early warning of impending no resources.</p> <p><b>Note 2:</b> The SONIC will stop reception of packets when the last RBA has been used and will not continue reception until additional receive buffers have been added (i.e., RWP is incremented beyond RRP) and this bit has been reset.</p> <p><b>Note 3:</b> If additional buffers have been added, resetting this bit causes the SONIC to read the next resource descriptor pointed to by the RRP in the Receive Resource Area. Note that resetting this bit under this condition is similar to issuing the Read RRA command (setting the RRRA bit in the Command Register). This bit should never be reset until after the additional resources have been added to the RRA.</p>
4	<p><b>RBAE: RECEIVE BUFFER AREA EXCEEDED</b></p> <p>Indicates that during reception, the SONIC has reached the end of the Receive Buffer Area. Reception is aborted and the SONIC fetches the next available resource descriptors in the RRA. The buffer space is not re-used and an RDA is not setup for the truncated packet (see Section 3.4.7).</p>
3	<p><b>CRC: CRC TALLY COUNTER ROLLOVER</b></p> <p>Indicates that the tally counter has rolled over from FFFFh to 0000h. (See Section 4.3.11.)</p>
2	<p><b>FAE: FRAME ALIGNMENT ERROR (FAE) TALLY COUNTER ROLLOVER</b></p> <p>Indicates that the FAE tally counter has rolled over from FFFFh to 0000h. (See Section 4.3.11.)</p>
1	<p><b>MP: MISSED PACKET (MP) COUNTER ROLLOVER</b></p> <p>Indicates that the MP tally counter has rolled over from FFFFh to 0000h. (See Section 4.3.11.)</p>
0	<p><b>RFO: RECEIVE FIFO OVERRUN</b></p> <p>Indicates that the SONIC has been unable to access the bus before the receive FIFO has filled from the network. This condition is due to excessively long bus latency and/or slow bus clock. Note that FIFO underruns are indicated in the TCR. (See Section 1.4.1.)</p>



## 4.0 SONIC Registers (Continued)

### 4.3.7 Data Configuration Register 2

(RA<5:0> = 3Fh)

This register (Figure 4-10) is for enabling the extended bus interface options.

A hardware reset will set all bits in this register to "0" except for the Extended Programmable Outputs which are unknown until written to and bits 5 to 11 which must always be written with 0s but are "don't cares" when read. A software reset will not affect any bits in this register. This register should only be written to when the SONIC is in software reset (the RST bit in the Command Register is set).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXPO3	EXPO2	EXPO1	EXPO0	0	0	0	0	0	0	0	PH	0	PCM	PCNM	RJCM
r/w	r/w	r/w	r/w								r/w		r/w	r/w	r/w

r/w = read/write

**FIGURE 4-10. Data Configuration Register**

Field	Meaning
EXPO3-0	EXTENDED PROGRAMMABLE OUTPUTS
PH	PROGRAM HOLD
PCM	PACKET COMPRESS WHEN MATCHED
PCNM	PACKET COMPRESS WHEN NOT MATCHED
RJCM	REJECT ON CAM MATCH

Bit	Description
15-12	<p><b>EXPO&lt;3:0&gt; EXTENDED PROGRAMMABLE OUTPUTS</b></p> <p>These bits program the level of the Extended User outputs (EXUSR&lt;3:0&gt;) when the SONIC is a bus master. Writing a "1" to any of these bits programs a high level to the corresponding output. Writing a "0" to any of these bits programs a low level to the corresponding output. EXUSR&lt;3:0&gt; are similar to USR&lt;1:0&gt; except that EXUSR&lt;3:0&gt; are only available when the Extended Bus mode is selected (bit 15 in the DCR is set to "1", see Section 4.3.2).</p>
11-5	Must be written with zeroes.
4	<p><b>PH: PROGRAM HOLD</b></p> <p>When this bit is set to "0", the HOLD request output is asserted/deasserted from the falling edge of bus clock. If this bit is set to "1", HOLD will be asserted/deasserted 1/2 clock later on the rising edge of bus clock.</p>
3	Must be zero.
2	<p><b>PCM: PACKET COMPRESS WHEN MATCHED</b></p> <p>When this bit is set to a "1" (and the PCNM bit is reset to a "0"), the <math>\overline{PCOMP}</math> output will be asserted if the destination address of the packet being received matches one of the entries in the CAM (Content Addressable Memory). This bit, along with PCNM, is used with the Management Bus of the DP83950, Repeater Interface Controller (RIC). See the DP83950 datasheet for more details on the RIC Management Bus. This mode is also called the Managed Bridge Mode.</p> <p><b>Note 1:</b> Setting PCNM and PCM to "1" at the same time is not allowed.</p> <p><b>Note 2:</b> If PCNM and PCM are both "0", the <math>\overline{PCOMP}</math> output will remain TRI-STATE until PCNM or PCM are changed.</p>
1	<p><b>PCNM: COMPRESS WHEN NOT MATCHED</b></p> <p>When this bit is set to a "1" (and the PCM bit is set to "0"), the <math>\overline{PCOMP}</math> output will be asserted if the destination address of the packet does not match one of the entries in the CAM. See the PCM bit above. This mode is also called the Managed Hub Mode.</p> <p><b>Note:</b> <math>\overline{PCOMP}</math> will not be asserted if the destination address is a broadcast address. This is true regardless of the state of the BRD bit in the Receive Control Register.</p>
0	<p><b>RJCM: REJECT ON CAM MATCH</b></p> <p>When this bit is set to "1", the SONIC will reject a packet on a CAM match. Setting RJCM to "0" causes the SONIC to operate normally by accepting packets on a CAM match. Setting this mode is useful for a small bridge with a limited number of nodes attached to it. RJCM only affects the CAM, though. Setting RJCM will not invert the function of the BRD, PRO or AMC bits (to accept broadcast, all physical or multicast packets respectively) in the Receive Control Register (see Section 4.3.3). This means, for example, that it is not possible to set RJCM and BRD to reject all broadcast packets. If RJCM and BRD are set at the same time, however, all broadcast packets will be accepted, but any packets that have a destination address that matches an address in the CAM will be rejected.</p>

## 4.0 SONIC Registers (Continued)

### 4.3.8 Transmit Registers

The transmit registers described in this section are part of the User Register set. The UTDA and CTDA must be initialized prior to issuing the transmit command (setting the TXP bit) in the Command register.

#### Upper Transmit Descriptor Address Register (UTDA):

This register contains the upper address bits ( $A<31:16>$ ) for accessing the transmit descriptor area (TDA) and is concatenated with the contents of the CTDA when the SONIC accesses the TDA in system memory. The TDA can be as large as 32k words or 16k long words and can be located anywhere in system memory. This register is unaffected by a hardware or software reset.

#### Current Transmit Descriptor Address Register (CTDA):

The 16-bit CTDA register contains the lower address bits ( $A<15:1>$ ) of the 32-bit transmit descriptor address. During initialization this register must be programmed with the lower address bits of the transmit descriptor. The SONIC concatenates the contents of this register with the contents of the UTDA to point to the transmit descriptor. For 32-bit memory systems bit 1, corresponding to address signal A1, must be set to "0" for alignment to long-word boundaries. Bit 0 of this register is the End of List (EOL) bit and is used to denote the end of the list. This register is unaffected by a hardware or software reset.

### 4.3.9 Receive Registers

The receive registers described in this section are part of the User Register set. A software reset has no effect on these registers and a hardware reset only affects the EOBC and RSC registers. The receive registers must be initialized prior to issuing the receive command (setting the RXEN bit) in the Command register.

#### Upper Receive Descriptor Address Register (URDA):

This register contains the upper address bits ( $A<31:16>$ ) for accessing the receive descriptor area (RDA) and is concatenated with the contents of the CRDA when the SONIC accesses the RDA in system memory. The RDA can be as large as 32k words or 16k long words and can be located anywhere in system memory. This register is unaffected by a hardware or software reset.

#### Current Receive Descriptor Address Register (CRDA):

The CRDA is a 16-bit read/write register used to locate the received packet descriptor block within the RDA. It contains the lower address bits ( $A<15:1>$ ). The SONIC concatenates the contents of the CRDA with the contents of the URDA to form the complete 32-bit address. The resulting 32-bit address points to the first field of the descriptor block. For 32-bit memory systems, bit 1, corresponding to address signal A1, must be set to "0" for alignment to long-word boundaries. Bit 0 of this register is the End of List (EOL) bit and is used to denote the end of the list. This register is unaffected by a hardware or software reset.

**End of Buffer Word Count Register (EOBC):** The SONIC uses the contents of this register to determine where to place the next packet. At the end of packet reception, the SONIC compares the contents of the EOBC register with the contents of the Remaining Buffer Word Count registers (RBWC0,1) to determine whether: (1) to place the next packet in the same RBA or (2) to place the next packet in another RBA. If the EOBC is less than or equal to the remaining number of words in the RBA after a packet is received (i.e.,  $EOB \leq RBWC0,1$ ), the SONIC buffers the next packet in the same RBA. If the EOBC is greater than

the remaining number of words in the RBA after a packet is received (i.e.,  $EOBC > RBWC0,1$ ), the Last Packet in RBA bit, LPKT in the Receive Control Register, Section 4.3.3, is set and the SONIC fetches the next resource descriptor. Hence, the next packet received will be buffered in a new RBA. A hardware reset sets this register to 02F8H (760 words or 1520 bytes). See Sections 3.4.2 and 3.4.4.4 for more information about using EOBC.

#### Upper Receive Resource Address Register (URRA):

The URRA is a 16-bit read/write register. It is programmed with the base address of the receive resource area (RRA). This 16-bit upper address value ( $A<31:16>$ ) locates the receive resource area in system memory. SONIC uses the URRA register when accessing the receive descriptors within the RRA by concatenating the lower address value from one of four receive resource registers (RSA, REA, RWP, or RRP).

#### Resource Start Address Register (RSA):

The RSA is a 15-bit read/write register. The LSB is not used and always reads back as a 0. The RSA is programmed with the lower 15-bits ( $A<15:1>$ ) of the starting address of the receive resource area. SONIC concatenates the contents of this register with the contents of the URRA to form the complete 32-bit address.

#### Resource End Address Register (REA):

The REA is a 15-bit read/write register. The LSB is not used and always reads back as a 0. The REA is programmed with the lower 15-bits ( $A<15:1>$ ) of the ending address of the receive resource area. SONIC concatenates the contents of this register with the contents of the URRA to form the complete 32-bit address.

#### Resource Read Pointer Register (RRP):

The RRP is a 15-bit read/write register. The LSB is not used and always reads back as a 0. The RRP is programmed with the lower 15-bit address ( $A<15:1>$ ) of the first field of the next descriptor the SONIC will read. SONIC concatenates the contents of this register with the contents of the URRA to form the complete 32-bit address.

#### Resource Write Pointer Register (RWP):

The RWP is a 15-bit read/write register. The LSB is not used and always reads back as a 0. The RWP is programmed with the lower 15-bit address ( $A<15:1>$ ) of the next available location the system can add a descriptor. SONIC concatenates the contents of this register with the contents of the URRA to form the complete 32-bit address. In 32-bit mode, bit 1, corresponding to address signal A1, must be zero to insure the proper equality comparison between this register and the RRP register.

#### Receive Sequence Counter Register (RSC):

This is a 16-bit read/write register containing two fields. The SONIC uses this register to provide status information on the number of packets within a RBA and the number of RBAs. The RSC register contains two 8-bit (modulo 256) counters. After each packet is received the packet sequence number is incremented. The SONIC maintains a single sequence number for each RBA. When the SONIC uses the next RBA, the packet sequence number is reset to zero and the RBA sequence number is incremented. This register is reset to 0 by a hardware reset or by writing zero to it. A software reset has no affect.

15	8	7	0
RBA Sequence Number (Modulo 256)		Packet Sequence Number (Modulo 256)	

## 4.0 SONIC Registers (Continued)

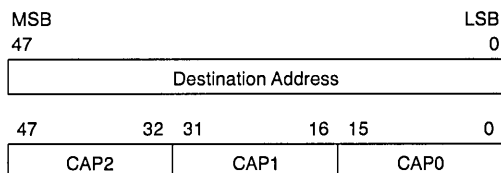
### 4.3.10 CAM Registers

The CAM registers described in this section are part of the User Register set. They are used to program the Content Addressable Memory (CAM) entries that provide address filtering of packets. These registers, except for the CAM Enable register, are unaffected by a hardware or software reset.

**CAM Entry Pointer Register (CEP):** The CEP is a 4-bit register used by SONIC to select one of the sixteen CAM entries. SONIC uses the least significant 4-bits of this register. The value of 0h points to the first CAM entry and the value of Fh points to the last entry.

**CAM Address Port 2, 1, 0 Registers (CAP2, CAP1, CAP0):** Each CAP is a 16-bit read-only register used to access the CAM cells. Each CAM cell is 16-bits wide and contains one third of the 48-bit CAM entry which is used by the SONIC for address filtering. The CAP2 register is used to access the upper bits (<47:32>), CAP1 the middle bits (<31:16>) and CAP0 the lower bits (<15:0>) of the CAM entry. Given the physical address 60:50:40:30:20:10, which is made up of 6 octets or bytes, where 10h is the least significant byte and 60h is the most significant byte (60h would be the first byte received from the network and 10h would be the last), CAP0 would be loaded with 2010h, CAP1 with 4030h and CAP2 with 6050h.

To read a CAM entry, the user first places the SONIC in software reset (set the RST bit in the Command register), programs the CEP register to select one of sixteen CAM entries, then reads CAP2, CAP1, and CAP0 to obtain the complete 48-bit entry. The user can not write to the CAM entries directly. Instead, the user programs the CAM descriptor area in system memory (see Section 4.1.1), then issues the Load CAM command (setting LCAM bit in the Command register). This causes the SONIC to read the descriptors from memory and loads the corresponding CAM entry through CAP2-0.



**CAM Enable Register (CE):** The CE is a 16-bit read/write register used to mask out or enable individual CAM entries. Each register bit position corresponds to a CAM entry. When a register bit is set to a "1" the corresponding CAM entry is enabled. When "0" the entry is disabled. This register is unaffected by a software reset and cleared to zero (disabling all entries) during a hardware reset. Under normal operations the user does not access this register. Instead the user sets up this register through the last entry in the CAM descriptor area. The SONIC loads the CE register during execution of the LCAM Command.

**CAM Descriptor Pointer Register (CDP):** The CDP is a 15-bit read/write register. The LSB is unused and always reads back as 0. The CDP is programmed with the lower

address (A<15:1>) of the first field of the CAM descriptor block in the CAM descriptor area (CDA) of system memory. SONIC uses the contents of the CDP register when accessing the CAM descriptors. This register must be programmed by the user before issuing the LCAM command. During execution of the LCAM Command SONIC concatenates the contents of this register with the contents of the URRR register to form the complete 32-bit address. During the Load CAM operation this register is incremented to address the fields in the CDA. After the Load Command completes, this register points to the next location after the CAM Descriptor Area.

**CAM Descriptor Count Register (CDC):** The CDC is a 5-bit read/write register. It is programmed with the number of CAM descriptor blocks in the CAM descriptor area. This register must be programmed by the user before issuing the LCAM command. SONIC uses the value in this register to determine how many entries to place in the CAM during execution of the LCAM command. During LCAM execution SONIC decrements this register each time it reads a descriptor block. When the CDC decrements to zero SONIC terminates the LCAM execution. Since the CDC register is programmed with the number of CAM descriptor blocks in the CAM Descriptor Area, the value programmed into the CDC register ranges 1 to 16 (1h to 10h).

### 4.3.11 Tally Counters

The SONIC provides three 16-bit counters used for monitoring network statistics on the number of CRC errors, Frame Alignment errors, and missed packets. These registers roll-over after the count of FFFFh is reached and produce an interrupt if enabled in the Interrupt Mask Register (IMR). These counters are unaffected by the RXEN bit in the CR, but are halted when the RST bit in the CR is set. The data written to these registers is inverted before being latched. This means that if a value of FFFFh is written to these registers by the system, they will contain and read back the value 0000h. Data is not inverted during a read operation. The Tally registers, therefore, are cleared by writing all "1's" to them. A software or hardware reset does not clear the tally counters.

**CRC Tally Counter Register (CRCT):** The CRCT is a 16-bit read/write register. This register is used to keep track of the number of packets received with CRC errors. After a packet is accepted by the address recognition logic, this register is incremented if a CRC error is detected. If the packet also contains a Frame Alignment error, this counter is not incremented.

**FAE Tally Counter Register (FAET):** The FAET is a 16-bit read/write register. This register is used to keep track of the number of packets received with frame alignment errors. After a packet is accepted by the address recognition logic, this register is incremented if a FAE error is detected.

**Missed Packet Tally Counter Register (MPT):** The MPT is a 16-bit read/write register. After a packet is received, this counter is incremented if there is: (1) lack of memory resources to buffer the packet, (2) a FIFO overrun, or (3) a valid packet has been received, but the receiver is disabled (RXDIS is set in the command register).

## 4.0 SONIC Registers (Continued)

### 4.3.12 General Purpose Timer

The SONIC contains a 32-bit general-purpose Watchdog Timer for timing user-definable events. This timer is accessed by the user through two 16-bit read/write registers (WT1 and WT0). The lower count value is programmed through the WT0 register and the upper count value is programmed through the WT1 register.

These two registers are concatenated together to form the complete 32-bit timer. This timer, clocked at  $\frac{1}{2}$  the Transmit Clock (TXC) frequency, counts down from its programmed value and generates an interrupt, if enabled (Interrupt Mask register), when it rolls over from 0000 0000h to FFFF FFFFh. When the counter rolls over it continues decrementing unless explicitly stopped (setting the STP bit). The timer is controlled by the ST (Start Timer) and STP (Stop Timer) bits in the Command register. A hardware or software reset halts, but does not clear, the General Purpose timer.

31	16	15	0
WT1 (Upper Count Value)		WT0 (Lower Count Value)	

### 4.3.13 Silicon Revision Register

This is a 16-bit read only register. It contains information on the current revision of the SONIC. The value of the DP83932CVF revision register is 6h.

## 5.0 Bus Interface

SONIC features a high speed non-multiplexed address and data bus designed for a wide range of system environments. The data bus can be programmed (via the Data Configuration Register) to a width of either 32- or 16-bits. SONIC con-

tains an on-chip DMA and supplies all the necessary signals for DMA operation. With 31 address lines SONIC can access a full 2 G-word address space. To accommodate different memory speeds wait states can be added to the bus cycle by two methods. The memory subsystem can add wait states by simply withholding the appropriate handshake signals. In addition, the SONIC can be programmed (via the Data Configuration Register) to add wait states.

The SONIC is designed to interface to both the National/Intel and Motorola style buses. To facilitate minimum chip count designs and complete bus compatibility the user can program the SONIC for the following bus modes:

- National/Intel bus operating in synchronous mode
- National/Intel bus operating in asynchronous mode
- Motorola bus operating in synchronous mode
- Motorola bus operating in asynchronous mode

The mode pin (BMODE) along with the SBUS bit in the Data Configuration Register are used to select the bus mode.

This section describes the SONIC's pin signals, provides system interface examples, and describes the various SONIC bus operations.

### 5.1 PIN CONFIGURATIONS

There are two user selectable pin configurations for SONIC to provide the proper interface signals for either the National/Intel or Motorola style buses. The state of the BMODE pin is used to define the pin configuration. *Figure 5-1* shows the pin configuration when BMODE = 1 (tied to  $V_{CC}$ ) for the Motorola style bus. *Figure 5-2* shows the pin configuration when BMODE = 0 (tied to ground) for the National/Intel style bus.

5.0 Bus Interface (Continued)

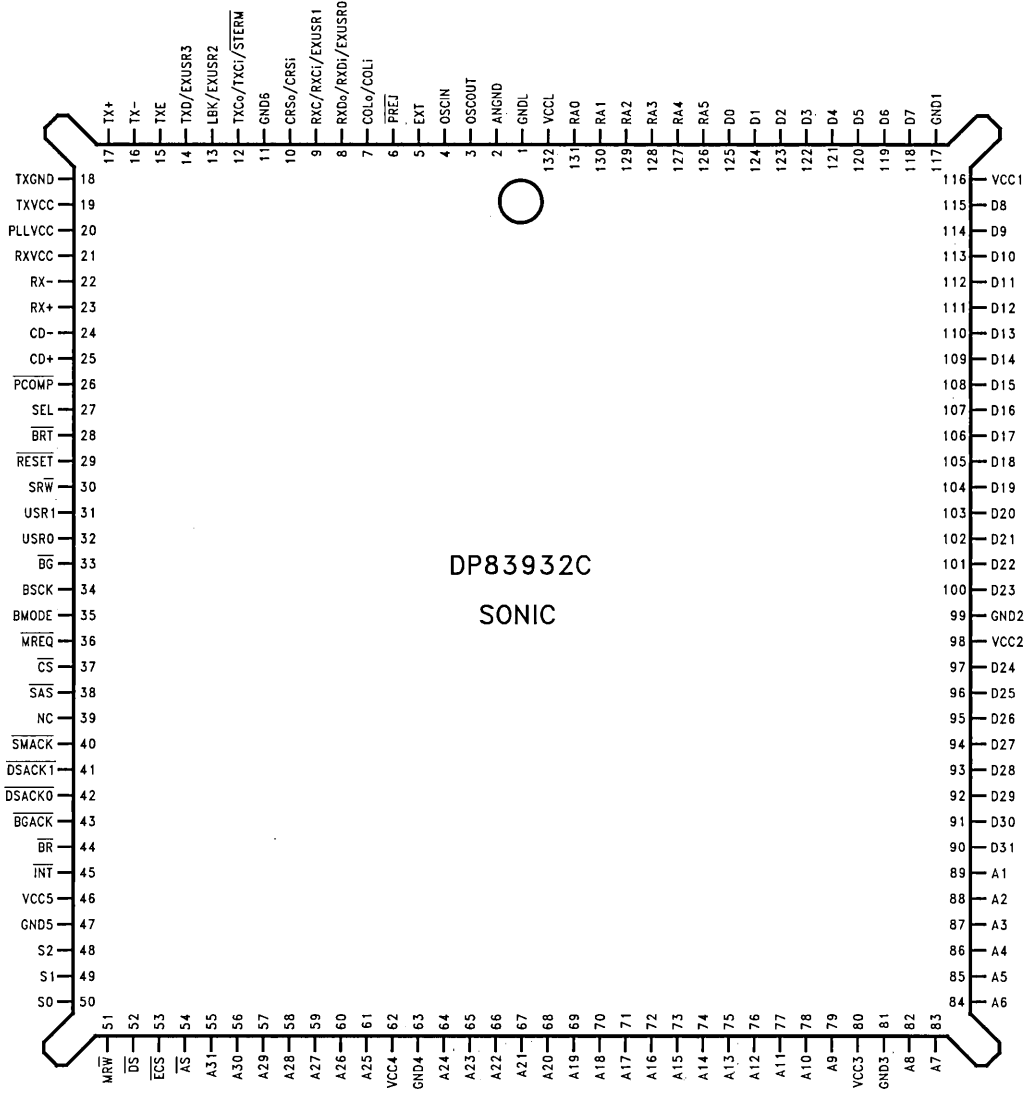
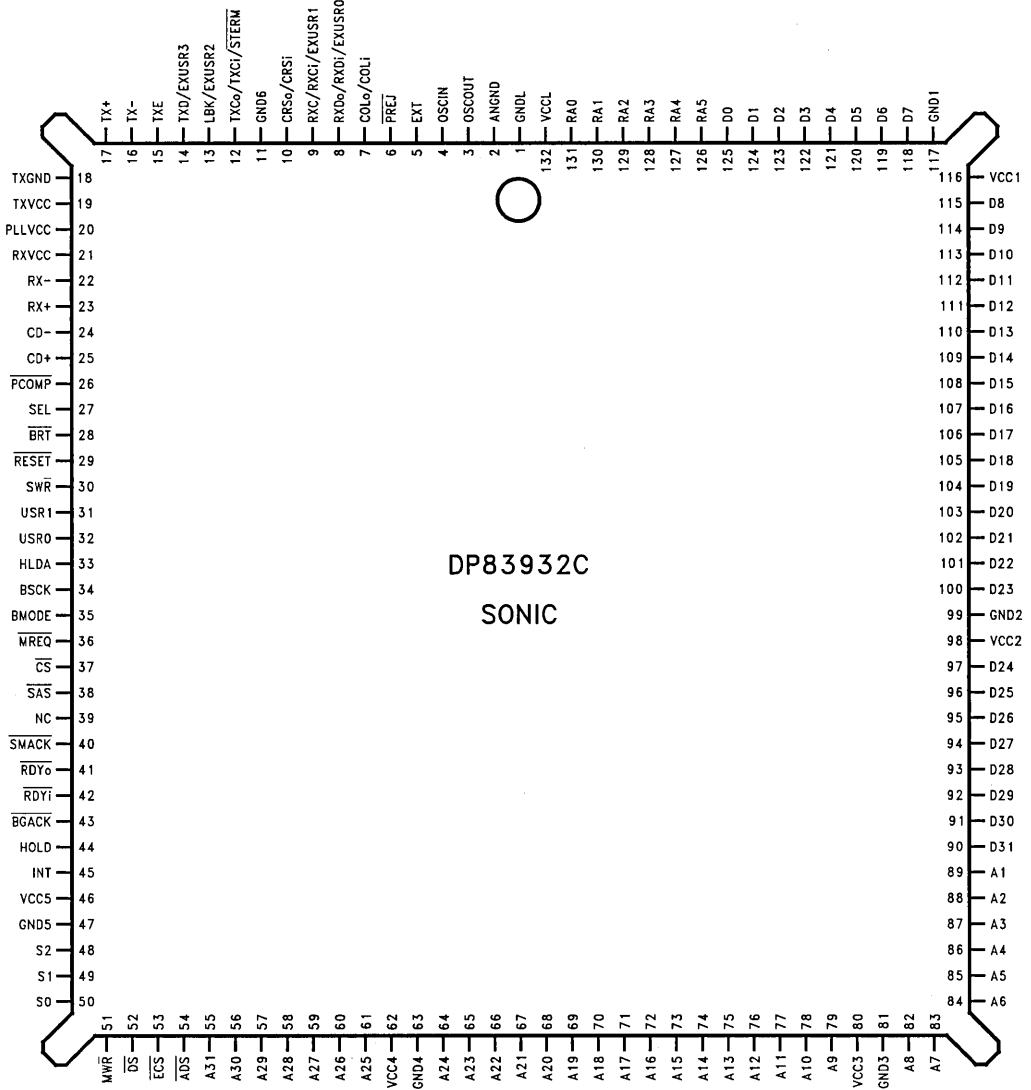


FIGURE 5-1. Connection Diagram (BMODE = 1)

TL/F/10492-23

5.0 Bus Interface (Continued)



DP83932C  
SONIC

FIGURE 5-2. Connection Diagram (BMODE = 0)

TL/F/10492-24

## 5.0 Bus Interface (Continued)

### 5.2 PIN DESCRIPTION

I = input

O = output

Z = TRI-STATE inputs are TTL compatible

ECL = ECL-like drivers for interfacing to the Attachment Unit Interface.

TP = Totem pole like drivers. These drivers are driven either high or low and are always driven. Drive levels are CMOS compatible.

TRI = TRI-STATE drivers. These pins are driven high, low or TRI-STATE. Drive levels are CMOS compatible. These pins may also be inputs (depending on the pin).

OC = Open Collector type drivers. These drivers are TRI-STATE when inactive and are driven low when active. These pins may also be inputs (depending on the pin). Pin names which contain a "/" indicate dual function pins.

TABLE 5-1. Pin Description

Symbol	Driver Type	Direction	Description
<b>NETWORK INTERFACE PINS</b>			
EXT		I	<b>External ENDEC Select:</b> Tying this pin to $V_{CC}$ (EXT = 1) disables the internal ENDEC and allows an external ENDEC to be used. Tying this pin to ground (EXT = 0) enables the internal ENDEC. This pin must be tied either to $V_{CC}$ or ground. Note the alternate pin definitions for CRS <sub>o</sub> /CRS <sub>i</sub> , COL <sub>o</sub> /COL <sub>i</sub> , RXD <sub>o</sub> /RXD <sub>i</sub> , RXC <sub>o</sub> /RXC <sub>i</sub> , and TXC <sub>o</sub> /TXC <sub>i</sub> . When EXT = 0 the first pin definition is used and when EXT = 1 the second pin definition is used.
CD+		I	<b>Collision +:</b> The positive differential collision input from the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
CD-		I	<b>Collision -:</b> The negative differential collision input from the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
RX+		I	<b>Receive +:</b> The positive differential receive data input from the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
RX-		I	<b>Receive -:</b> The negative differential receive data input from the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
TX+	ECL	O	<b>Transmit +:</b> The positive differential transmit output to the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
TX-	ECL	O	<b>Transmit -:</b> The negative differential transmit output to the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
CRS <sub>o</sub> / CRS <sub>i</sub>	TP	O I	<b>Carrier Sense Output (CRS<sub>o</sub>)</b> from the internal ENDEC (EXT = 0): When EXT = 0 the CRS <sub>o</sub> signal is internally connected between the ENDEC and MAC units. It is asserted on the first valid high-to-low transition in the receive data (RX + / -). This signal remains active 1.5 bit times after the last bit of data. Although this signal is used internally by the SONIC it is also provided as an output to the user. <b>Carrier Sense Input (CRS<sub>i</sub>)</b> from an external ENDEC (EXT = 1): The CRS <sub>i</sub> signal is activated high when the external ENDEC detects valid data at its receive inputs.
COL <sub>o</sub> / COL <sub>i</sub>	TP	O I	<b>Collision Output (COL<sub>o</sub>)</b> from the internal ENDEC (EXT = 0): When EXT = 0 the COL <sub>o</sub> signal is internally connected between the ENDEC and MAC units. This signal generates an active high signal when the 10 MHz collision signal from the transceiver is detected. Although this signal is used internally by the SONIC it is also provided as an output to the user. <b>Collision Detect Input (COL<sub>i</sub>)</b> from an external ENDEC (EXT = 1): The COL <sub>i</sub> signal is activated from an external ENDEC when a collision is detected. This pin is monitored during transmissions from the beginning of the Start Of Frame Delimiter (SFD) to the end of the packet. At the end of transmission, this signal is monitored by the SONIC for CD heartbeat.

## 5.0 Bus Interface (Continued)

TABLE 5-1. Pin Description (Continued)

Symbol	Driver Type	Direction	Description
<b>NETWORK INTERFACE PINS (Continued)</b>			
RXDo/ RXDi/ EXUSR0	TP TRI	O I O, Z	<p>This pin will be TRI-STATE until the DCR has been written to. (See Section 4.3.2, EXBUS, for more information.)</p> <p><b>Receive Data Output (RXDo)</b> from the internal ENDEC (EXT = 0): NRZ data output. When EXT = 0 the RXDOUT signal is internally connected between the ENDEC and MAC units. This signal must be sampled on the rising edge of the receive clock output (RXCo). Although this signal is used internally by the SONIC it is also provided as an output to the user.</p> <p><b>Receive Data Input (RXDi)</b> from an external ENDEC (EXT = 1): The NRZ data decoded from the external ENDEC. This data is clocked in on the rising edge of RXCi.</p> <p><b>Extended User Output (EXUSR0):</b> When EXBUS has been set (see Section 4.3.2), this pin becomes a programmable output. It will remain TRI-STATE until the SONIC becomes a bus master, at which time it will be driven according to the value programmed in the DCR2 (Section 4.3.7).</p>
RXCo/ RXCi/ EXUSR1	TP TRI	O I O, Z	<p>This pin will be TRI-STATE until the DCR has been written to. (See Section 4.3.2, EXBUS, for more information.)</p> <p><b>Receive Clock Output (RXCo)</b> from the internal ENDEC (EXT = 0): When EXT = 0 the RXCo signal is internally connected between the ENDEC and MAC units. This signal is the separated receive clock from the Manchester data stream. It remains active 5-bit times after the deassertion of CRS0. Although this signal is used internally by the SONIC it is also provided as an output to the user.</p> <p><b>Receive Clock Input (RXCi)</b> from an external ENDEC (EXT = 1): The separated received clock from the Manchester data stream. This signal is generated from an external ENDEC.</p> <p><b>Extended User Output (EXUSR1):</b> When EXBUS has been set (see Section 4.3.2), this pin becomes a programmable output. It will remain TRI-STATE until the SONIC becomes a bus master, at which time it will be driven according to the value programmed in the DCR2 (Section 4.3.7).</p>
TXD/ EXUSR3	TP TRI	O O, Z	<p>This pin will be TRI-STATE until the DCR has been written to. (See Section 4.3.2, EXBUS, for more information.)</p> <p><b>Transmit Data (TXD):</b> The serial NRZ data from the MAC unit which is to be decoded by an external ENDEC. Data is valid on the rising edge of TXC. Although this signal is used internally by the SONIC it is also provided as an output to the user.</p> <p><b>Extended User Output (EXUSR3):</b> When EXBUS has been set (see Section 4.3.2), this pin becomes a programmable output. It will remain TRI-STATE until the SONIC becomes a bus master, at which time it will be driven according to the value programmed in the DCR2 (Section 4.3.7).</p>
TXE	TP	O	<p><b>Transmit Enable:</b> This pin is driven high when the SONIC begins transmission and remains active until the last byte is transmitted. Although this signal is used internally by the SONIC it is also provided as an output to the user.</p>
TXCo/ TXCi/ STERM	TRI	O, Z I I	<p>This pin will be TRI-STATE until the DCR has been written to. (See Section 4.3.2, EXBUS, for more information.)</p> <p><b>Transmit Clock Output (TXCo)</b> from the internal ENDEC (EXT = 0): This 10 MHz clock transmit clock output is derived from the 20 MHz oscillator. When EXT = 0 the TXCOUT signal is internally connected between the ENDEC and MAC units. Although this signal is used internally by the SONIC it is also provided as an output to the user.</p> <p><b>Transmit Clock Input (TXCi)</b> (EXT = 1): This input clock from an external ENDEC is used for shifting data out of the MAC unit serializer. This clock is nominally 10 MHz.</p> <p><b>Synchronous Termination (STERM):</b> When the SONIC is a bus master, it samples this pin before terminating its memory cycle. This pin is sampled synchronously and may only be used in asynchronous bus mode when BMODE = 1. See Section 5.4.5 for more details.</p>



## 5.0 Bus Interface (Continued)

TABLE 5-1. Pin Description (Continued)

Symbol	Driver Type	Direction	Description
<b>NETWORK INTERFACE PINS</b> (Continued)			
LBK/ EXUSR2	TP TRI	O O, Z	This pin will be TRI-STATE until the DCR has been written to. (See Section 4.3.2, EXBUS, for more information.) <b>Loopback (LBK):</b> When ENDEC loopback is programmed, LBK is asserted high. Although this signal is used internally by the SONIC it is also provided as an output to the user. <b>Extended User Output (EXUSR2):</b> When EXBUS has been set (see Section 4.3.2), this pin becomes a programmable output. It will remain TRI-STATE until the SONIC becomes a bus master, at which time it will be driven according to the value programmed in the DCR2 (Section 4.3.7).
PCOMP	TRI	O, Z	<b>Packet Compression:</b> This pin is used with the Management Bus of the DP83950, Repeater Interface Controller (RIC). The SONIC can be programmed to assert PCOMP whenever there is a CAM match, or when there is not a match. The RIC uses this signal to compress (shorten) a received packet for management purposes and to reduce memory usage. (See the DP83950 datasheet for more details on the RIC Management Bus.) The operation of this pin is controlled by bits 1 and 2 in the DCR2 register. PCOMP will remain TRI-STATE until these bits are written to. This signal is asserted right after the 4th bit of the 7th byte of the incoming packet and is deasserted one transmit clock (TXC) after CRS is driven low.
SEL		I	<b>Mode Select (EXT = 0):</b> This pin is used to determine the voltage relationship between TX+ and TX- during idle at the primary of the isolation transformer on the network interface. When tied to V <sub>CC</sub> , TX+ and TX- are at equal voltages during idle. When tied to ground, the voltage at TX+ is positive with respect to TX- during idle on the primary side of the isolation transformer ( <i>Figure 6-2</i> ).
PREJ		I	<b>Packet Reject:</b> This signal is used to reject received packets. When asserted low for at least two receive clocks (RXC), the SONIC will reject the incoming packet. This pin can be asserted up to the 2nd to the last bit of reception to reject a packet.
OSCOU	TP	O	<b>Crystal Feedback Output:</b> This signal is used to provide clocking signals for the internal ENDEC. A crystal can be connected to this pin along with OSCIN. See Section 6.1.3 for more information about using oscillators or crystals.
OSCIN		I	<b>Crystal Feedback Input or External Oscillator Input:</b> This signal is used to provide clocking signals for the internal ENDEC. A crystal may be connected to this pin along with OSCOUT, or an oscillator module may be used. Typically the output of an oscillator module is connected to this pin. See Section 6.1.3 for more information about using oscillator modules or crystals.
<b>BUS INTERFACE PINS</b>			
BMODE		I	<b>Bus Mode:</b> This input enables the SONIC to be compatible with standard microprocessor buses. The level of this pin affects byte ordering (little or big endian) and controls the operation of the bus interface control signals. A high level (tied to V <sub>CC</sub> ) selects Motorola mode (big endian) and a low level (tied to ground) selects National/Intel mode (little endian). Note the alternate pin definitions for AS/ADS, MRW/MWR, INT/INT, BR/HOLD, BG/HLDA, SRW/SWR, DSACK0/RDY0, and DSACK1/RDY1.

## 5.0 Bus Interface (Continued)

TABLE 5-1. Pin Description (Continued)

Symbol	Driver Type	Direction	Description
<b>BUS INTERFACE PINS (Continued)</b>			
D31–D0	TRI	I, O, Z	<b>Data Bus:</b> These bidirectional lines are used to transfer data on the system bus. When the SONIC is a bus master, 16-bit data is transferred on D15–D0 and 32-bit data is transferred on D31–D0. When the SONIC is accessed as a slave, register data is driven onto lines D15–D0. D31–D16 are held TRI-STATE if SONIC is in 16-bit mode. If SONIC is in 32-bit mode, they are driven, but invalid.
A31–A1	TRI	O, Z	<b>Address Bus:</b> These signals are used by the SONIC to drive the DMA address after the SONIC has acquired the bus. Since the SONIC aligns data to word boundaries, only 31 address lines are needed.
RA5–RA0		I	<b>Register Address Bus:</b> These signals are used to access SONIC's internal registers. When the SONIC is accessed, the CPU drives these lines to select the desired SONIC register.
$\overline{AS}$ / $\overline{ADS}$	TRI TRI	O, Z O, Z	<b>Address Strobe (<math>\overline{AS}</math>):</b> When BMODE = 1, the falling edge indicates valid status and address. The rising edge indicates the termination of the memory cycle. <b>Address Strobe (<math>\overline{ADS}</math>):</b> When BMODE = 0, the rising edge indicates valid status and address.
$\overline{MRW}$ / $\overline{MWR}$	TRI TRI	O, Z O, Z	When the SONIC has acquired the bus, this signal indicates the direction of data. <b>Memory Read/Write Strobe (<math>\overline{MRW}</math>):</b> When BMODE = 1, this signal is high during a read cycle and low during a write cycle. <b>Memory Read/Write Strobe (<math>\overline{MWR}</math>):</b> When BMODE = 0, the signal is low during a read cycle and high during a write cycle.
$\overline{INT}$ / INT	OC TP	O, Z O	Indicates that an interrupt (if enabled) is pending from one of the sources indicated by the Interrupt Status register. Interrupts that are disabled in the Interrupt Mask register will not activate this signal. <b>Interrupt (<math>\overline{INT}</math>):</b> This signal is active low when BMODE = 1. <b>Interrupt (INT):</b> This signal is active high when BMODE = 0.
$\overline{RESET}$		I	<b>Reset:</b> This signal is used to hardware reset the SONIC. When asserted low, the SONIC transitions into the reset state after 10 transmit clocks or 10 bus clocks if the bus clock period is greater than the transmit clock period.
S2–S0	TP	O	<b>Bus Status:</b> These three signals provide a continuous status of the current SONIC bus operations. See Section 5.4.3 for status definitions.
BCLK		I	<b>Bus Clock:</b> This clock provides the timing for the SONIC DMA engine.
$\overline{BR}$ / HOLD	OC TP	O, Z O	<b>Bus Request (<math>\overline{BR}</math>):</b> When BMODE = 1, the SONIC asserts this pin low when it attempts to gain access to the bus. When inactive this signal is tri-stated. <b>Hold Request (HOLD):</b> When BMODE = 0, the SONIC drives this pin high when it intends to use the bus and is driven low when inactive.
$\overline{BG}$ / HLDA		I I	<b>Bus Grant (<math>\overline{BG}</math>):</b> When BMODE = 1 this signal is a bus grant. The system asserts this pin low to indicate potential mastership of the bus. <b>Hold Acknowledge (HLDA):</b> When BMODE = 0 this signal is used to inform the SONIC that it has attained the bus. When the system asserts this pin high, the SONIC has gained ownership of the bus. This signal is sampled synchronously and the setup time must be met to ensure proper operation.
$\overline{BGACK}$	TRI	I, O, Z	<b>Bus Grant Acknowledge:</b> When BMODE = 1, the SONIC asserts this pin low when it has determined that it can gain ownership of the bus. The SONIC checks the following signal before driving $\overline{BGACK}$ . 1) $\overline{BG}$ has been received through the bus arbitration process. 2) $\overline{AS}$ is deasserted, indicating that the CPU has finished using the bus. 3) $\overline{DSACK0}$ and $\overline{DSACK1}$ are deasserted, indicating that the previous slave device is off the bus. 4) $\overline{BGACK}$ is deasserted, indicating that the previous master is off the bus. This pin is only used when BMODE = 1.

## 5.0 Bus Interface (Continued)

TABLE 5-1. Pin Description (Continued)

Symbol	Driver Type	Direction	Description
<b>BUS INTERFACE PINS (Continued)</b>			
$\overline{CS}$		I	<p><b>Chip Select:</b> The system asserts this pin low to access the SONIC's registers. The registers are selected by placing an address on lines RA5–RA0.</p> <p><b>Note:</b> Both <math>\overline{CS}</math> and <math>\overline{MREQ}</math> must not be asserted concurrently. If these signals are successively asserted, there must be at least two bus clocks between the deasserting edge of the first signal and the asserting edge of the second signal.</p>
$\overline{SAS}$		I	<p><b>Slave Address Strobe:</b> The system asserts this pin to latch the register address on lines RA0–RA5. When BMODE = 1, the address is latched on the falling edge of <math>\overline{SAS}</math>. When BMODE = 0 the address is latched on the rising edge of <math>\overline{SAS}</math>.</p>
SRW/ SWR		I I	<p>The system asserts this pin to indicate whether it will read from or write to the SONIC's registers.</p> <p><b>Slave Read/Write (SRW):</b> When BMODE = 1, this signal is asserted high during a read and low during a write.</p> <p><b>Slave Write/Read (SWR):</b> when BMODE = 0, this signal is asserted low during a read and high during a write.</p>
$\overline{DS}$	TRI	O, Z	<p><b>Data Strobe:</b> When the SONIC is bus master, it drives this pin low during a read cycle to indicate that the slave device may drive data onto the bus; in a write cycle, this pin indicates that the SONIC has placed valid data onto the bus.</p>
$\overline{DSACK0}/$ $\overline{RDYi}/$ $\overline{DSACK1}/$ $\overline{RDY0}$	TRI TRI TP	I, O, Z I I, O, Z O	<p><b>Data and Size Acknowledge 0 and 1 (<math>\overline{DSACK0,1}</math> BMODE = 1):</b> These pins are the output slave acknowledge to the system when the SONIC registers have been accessed and the input slave acknowledgement when the SONIC is busmaster. When a register has been accessed, the SONIC drives both <math>\overline{DSACK0}</math> and <math>\overline{DSACK1}</math> pins low to terminate the slave cycle. (Note that the SONIC responds as a 32-bit peripheral by driving both <math>\overline{DSACK0}</math> and <math>\overline{DSACK1}</math> low, but drives data only on lines D0–D15. Lines D16–D31 are driven, but invalid.) When the SONIC is bus master, it samples these pins before terminating its memory cycle. When SONIC is in 32-bit bus master mode, both <math>\overline{DSACK0}</math> and <math>\overline{DSACK1}</math> must be asserted to terminate the cycle. However, if the SONIC is in 16-bit bus master mode, only the assertion of <math>\overline{DSACK1}</math> is required to terminate the cycle. These pins are sampled synchronously or asynchronously depending on the state of the SBUS bit in the Data Configuration register. See Section 5.4.5 for details. Note that the SONIC does not allow dynamic bus sizing. Bus size is statically defined in the Data Configuration register (see Section 4.3.2).</p> <p><b>Ready Input (RDYi, BMODE = 0):</b> When the SONIC is a bus master, the system asserts this signal high to insert wait-states and low to terminate the memory cycle. This signal is sampled synchronously or asynchronously depending on the state of the SBUS bit. See Section 5.4.5 and 4.3.2 for details.</p> <p><b>Ready Output (<math>\overline{RDY0}</math>, BMODE = 0):</b> When a register is accessed, the SONIC asserts this signal to terminate the slave cycle.</p>
BRT		I	<p><b>Bus Retry:</b> When the SONIC is bus master, the system asserts this signal to rectify a potentially correctable bus error. This pin has 2 modes. Mode 1 (the LBR in the Data Configuration register is set to 0): Assertion of this pin forces the SONIC to terminate the current bus cycle and will repeat the same cycle after BRT has been deasserted. Mode 2 (the LBR bit in the Data Configuration register is set to 1): Assertion of this signal forces the SONIC to retry the bus operation as in Mode 1. However, the SONIC will not continue DMA operations until the BR bit in the ISR is reset.</p>
ECS	TRI	O, Z	<p><b>Early Cycle Start:</b> This output gives the system earliest indication that a memory operation is occurring. This signal is driven low at the rising edge of T1 and high at the falling edge of T1.</p>

## 5.0 Bus Interface (Continued)

**TABLE 5-1. Pin Description (Continued)**

Symbol	Driver Type	Direction	Description
<b>SHARED-MEMORY ACCESS PINS</b>			
$\overline{\text{MREQ}}$		I	<p><b>Memory Request:</b> The system asserts this signal low when it attempts to access the shared-buffer RAM. The on-chip arbiter resolves accesses between the system and the SONIC.</p> <p><b>Note:</b> Both <math>\overline{\text{CS}}</math> and <math>\overline{\text{MREQ}}</math> must not be asserted concurrently. If these signals are successively asserted, there must be at least two bus clocks between the deasserting edge of the first signal and the asserting edge of the second signal.</p> <p>In Motorola mode, if a bus master uses the <math>\overline{\text{MREQ}}</math> to request the bus from the SONIC, care should be taken to isolate the <math>\overline{\text{DSACK0,1}}</math> from the bus (e.g., use tri-state buffers) because the <math>\overline{\text{DSACK0,1}}</math> will be driven by the SONIC even after the SONIC has given up the bus.</p>
$\overline{\text{SMACK}}$	TP	O	<p><b>Slave and Memory Acknowledge:</b> SONIC asserts this dual function pin low in response to either a Chip Select (<math>\overline{\text{CS}}</math>) or a Memory Request (<math>\overline{\text{MREQ}}</math>) when the SONIC's registers or its buffer memory is available for accessing. This pin can be used for enabling bus drivers for dual-bus systems.</p>
<b>USER DEFINABLE PINS</b>			
USR0,1	TRI	I, O, Z	<p><b>User Define 0,1:</b> These signals are inputs when SONIC is hardware reset and are outputs when SONIC is a bus master (HLDA or <math>\overline{\text{BGACK}}</math> asserted). When hard reset (<math>\overline{\text{RST}}</math>) is low, these signals input directly into bits 8 and 9 of the Data Configuration register (DCR) respectively. The levels on these pins are latched on the rising edge of <math>\overline{\text{RST}}</math>. During busmaster operations (HLDA or <math>\overline{\text{BGACK}}</math> is active), these pins are outputs whose levels are programmable through bits 11 and 12 of the DCR respectively. The USR0,1 pins should be pulled up to <math>V_{\text{CC}}</math> or pulled down to ground. A 4.7 k<math>\Omega</math> pull-up resistor is recommended.</p>
<b>POWER AND GROUND PINS</b>			
VCC1-5 VCCL			<p><b>Power:</b> The +5V power supply for the digital portions of the SONIC.</p>
TXVCC RXVCC PLLVCC			<p><b>Power:</b> These pins are the +5V power supply for the SONIC ENDEC unit. These pins must be tied to <math>V_{\text{CC}}</math> even if the internal ENDEC is not used.</p>
GND1-6 GNDL			<p><b>Ground:</b> The ground reference for the digital portions of the SONIC.</p>
TXGND ANGND			<p><b>Ground:</b> These pins are the ground references for the SONIC ENDEC unit. These pins must be tied to ground even if the internal ENDEC is not used.</p>

### 5.3 SYSTEM CONFIGURATION

Any device that meets the SONIC interface protocol and electrical requirements (timing, threshold, and loading) can be interfaced to SONIC. Since two bus protocols are provided, via the BMODE pin, the SONIC can interface directly to most microprocessors. *Figure 5-3* shows a typical interface to the National/Intel style bus (BMODE=0) and *Figure 5-4* shows a typical interface to the Motorola style bus (BMODE=1).

The BMODE pin also controls byte ordering. When BMODE=1 big endian byte ordering is selected and when BMODE=0 little endian byte ordering is selected.

### 5.4 BUS OPERATIONS

There are two types of system bus operations: 1) SONIC as a slave, and 2) SONIC as a bus master. When SONIC is a slave (e.g., a CPU accessing SONIC registers) all transfers are non-DMA. When SONIC is a bus master (e.g., SONIC accessing receive or transmit buffer/descriptor areas) all transfers are block transfers using SONIC's on-chip DMA. This section describes the SONIC bus operations. Pay special attention to all sections labeled as "**Note**". These conditions must be met for proper bus operation.

5.0 Bus Interface (Continued)

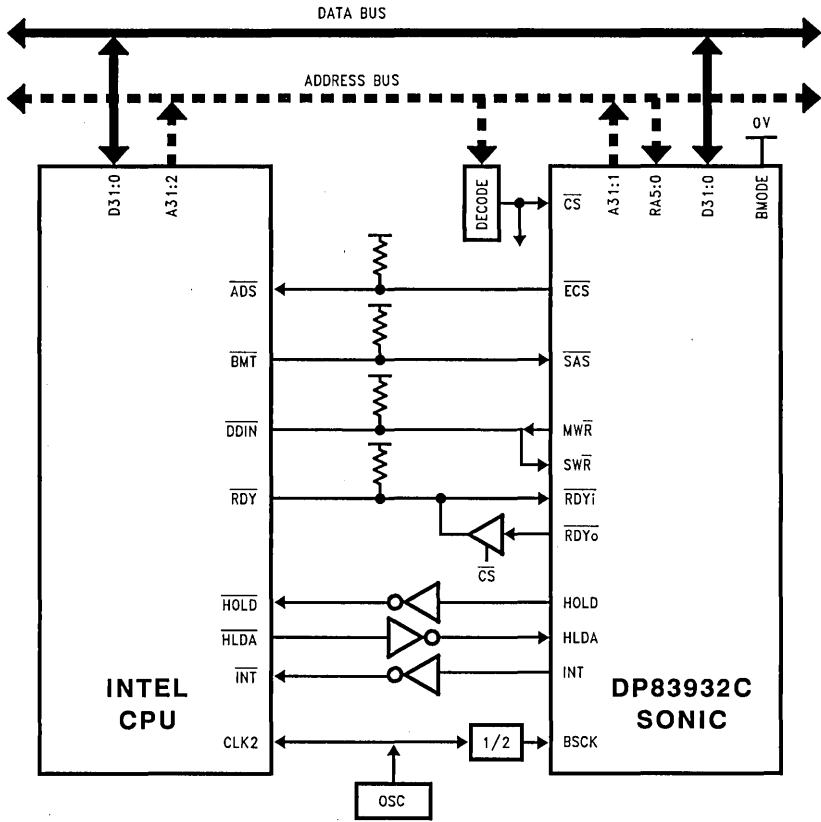


FIGURE 5-3. SONIC to Intel CPU Interface Example

TL/F/10492-25

5.0 Bus Interface (Continued)

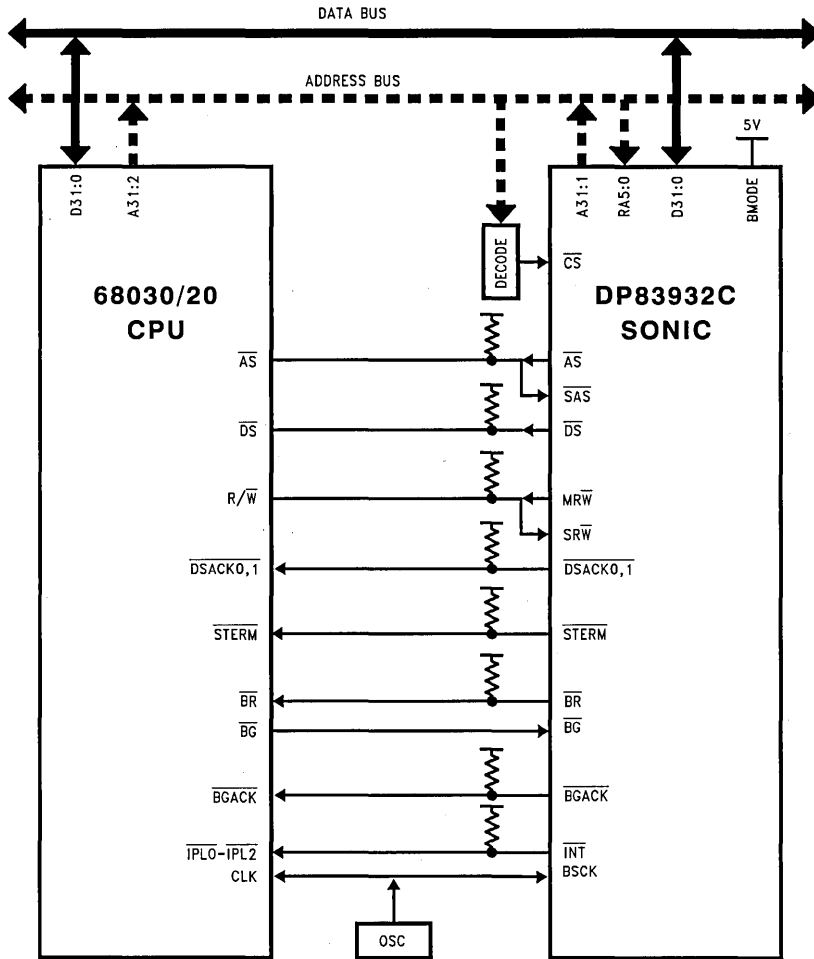


FIGURE 5-4. SONIC to Motorola 68030/20 Interface Example

TL/F/10492-26

## 5.0 Bus Interface (Continued)

### 5.4.1 Acquiring The Bus

The SONIC requests the bus when 1) its FIFO threshold has been reached or 2) when the descriptor areas in memory (i.e., RRA, RDA, CDA, and TDA) are accessed. Note that when the SONIC moves from one area in memory to another (e.g., RBA to RDA), it always deasserts its bus request and then requests the bus again when accessing the next area in memory.

The SONIC provides two methods to acquire the bus for compatibility with National/Intel or Motorola type microprocessors. These two methods are selected by setting the proper level on the BMODE pin.

Figures 5-5 and 5-6 show the National/Intel (BMODE = 0) and Motorola (BMODE = 1) bus request timing. Descriptions of each mode follows. For both modes, when the SONIC relinquishes the bus, there is an extra holding state (Th) for one bus cycle after the last DMA cycle (T2). This assures that the SONIC does not contend with another bus master after it has released the bus.

#### BMODE = 0

The National/Intel processors require a 2-way handshake using a HOLD REQUEST/HOLD ACKNOWLEDGE protocol (Figure 5-5). When the SONIC needs to access the bus, it issues a HOLD REQUEST (HOLD) to the microprocessor. The microprocessor, responds with a HOLD ACKNOWLEDGE (HLDA) to the SONIC. The SONIC then begins its memory transfers on the bus. As long as the CPU maintains HLDA active, the SONIC continues until it has finished its memory block transfer. The CPU, however, can preempt the SONIC from finishing the block transfer by deasserting HLDA before the SONIC deasserts HOLD. This allows a higher priority device to preempt the SONIC from continuing to use the bus. The SONIC will request the bus again later to complete any operation that it was doing at the time of preemption. The HLDA signal is sampled synchronously by the SONIC at the rising edge of the BSCCK, setup time must be met to ensure proper operation.

As shown in Figure 5-5, the SONIC will assert HOLD to either the falling or rising edge of the bus clock (BSCCK). The default is for HOLD to be asserted on the falling edge. Setting the PH bit in the DCR2 (see Section 4.3.7) causes HOLD to be asserted  $\frac{1}{2}$  bus clock later on the rising edge (shown by the dotted line). Before HOLD is asserted, the SONIC checks the HLDA line. If HLDA is asserted, HOLD will not be asserted until after HLDA has been deasserted first.

**Note:** If HLDA is driven low to preempt the SONIC from the bus while the SONIC is accessing the CAM (LCAM command), the SONIC will get off the bus but will not deassert HOLD even though the status bit will indicate idle state. If HLDA is driven low while the SONIC is accessing descriptor areas (RRA, RDA, TDA), the SONIC will be preempted normally (i.e., get off the bus and deassert HOLD) and the HOLD signal will be reasserted again after one bus clock. If HLDA is driven low while the SONIC is accessing data areas (RBA, TBA), the SONIC will be preempted normally but may not reassert HOLD unless required to do so depending on the threshold condition of the FIFO.

#### BMODE = 1

The Motorola protocol requires a 3-way handshake using a BUS REQUEST, BUS GRANT, and BUS GRANT ACKNOWLEDGE handshake (Figure 5-6). When using this protocol, the SONIC requests the bus by lowering BUS REQUEST ( $\overline{BR}$ ). The CPU responds by issuing BUS GRANT ( $\overline{BG}$ ). Upon receiving  $\overline{BG}$ , the SONIC assures that all devices have relinquished control of the bus before using the bus. The following signals must be deasserted before the SONIC acquires the bus:

- $\overline{BGACK}$
- $\overline{AS}$
- $\overline{DSACK0,1}$
- STERM (Asynchronous Mode Only)

Deasserting  $\overline{BGACK}$  indicates that the previous master has released the bus. Deasserting  $\overline{AS}$  indicates that the previous master has completed its cycle and deasserting  $\overline{DSACK0,1}$  and STERM indicates that the previous slave has terminated its connection to the previous master. The SONIC maintains its mastership of the bus until it deasserts  $\overline{BGACK}$ . It cannot be preempted from the bus.

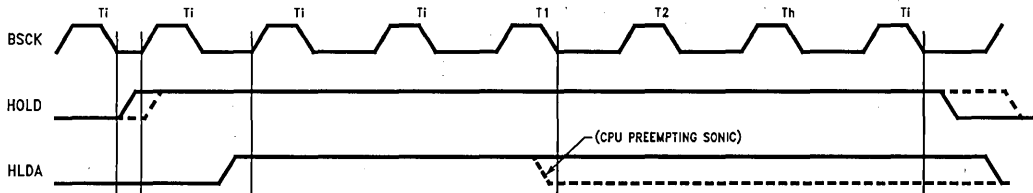


FIGURE 5-5. Bus Request Timing, BMODE = 0

TL/F/10492-27

## 5.0 Bus Interface (Continued)

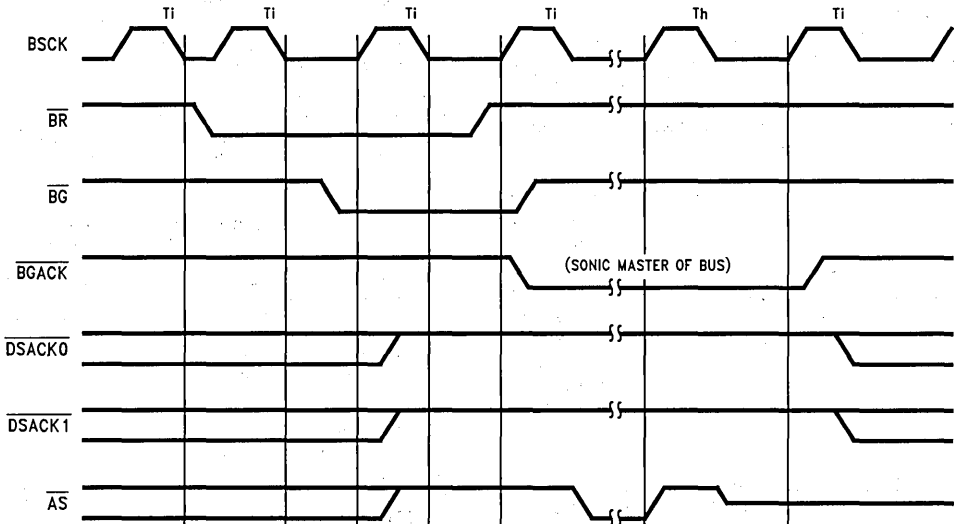


FIGURE 5-6. Bus Request Timing BMODE = 1

TL/F/10492-28

### 5.4.2 Block Transfers

The SONIC performs block operations during all bus actions, thereby providing efficient transfers to memory. The block cycle consists of three parts. The first part is the bus acquisition phase, as discussed above, in which the SONIC gains access to the bus. Once it has access of the bus, the SONIC enters the second phase by transferring data to/from its internal FIFOs or registers from/to memory. The SONIC transfers data from its FIFOs in either EXACT BLOCK mode or EMPTY/FILL mode.

**EXACT BLOCK mode:** In this mode the number of words (or long words) transferred during a block transfer is determined by either the Transmit or Receive FIFO thresholds programmed in the Data Configuration Register.

**EMPTY/FILL mode:** In this mode the DMA completely fills the Transmit FIFO during transmission, or completely empties the Receive FIFO during reception. This allows for greater bus latency.

When the SONIC accesses the Descriptor Areas (i.e., RRA, RDA, CDA, and TDA), it transfers data between its registers and memory. All fields which need to be used are accessed in one block operation. Thus, the SONIC performs 4 accesses in the RRA (see Section 3.4.4.2), 7 accesses in the RDA (see Section 3.4.6.1), 2, 3, or 6 accesses in the TDA (see Section 3.5.4) and 4 accesses in the CDA.

### 5.4.3 Bus Status

The SONIC presents three bits of status information on pins S2-S0 which indicate the type of bus operation the SONIC is currently performing (Table 5-2). Bus status is valid at the falling edge of AS or the rising edge of ADS.

TABLE 5-2. Bus Status

S2	S1	S0	Status
1	1	1	The bus is idle. The SONIC is not performing any transfers on the bus.
1	0	1	The Transmit Descriptor Area (TDA) is currently being accessed.
0	0	1	The Transmit Buffer Area (TBA) is currently being read.
0	1	1	The Receive Buffer Area (RBA) is currently being written to. Only data is being written, though, not a Source or Destination address.
0	1	0	The Receive Buffer Area (RBA) is currently being written to. Only the Source or Destination address is being written though.
1	1	0	The Receive Resource Area (RRA) is currently being read.
1	0	0	The Receive Descriptor Area (RDA) is currently being accessed.
0	0	0	The CAM Descriptor Area (CDA) is currently being accessed.



## 5.0 Bus Interface (Continued)

### 5.4.3.1 Bus Status Transitions

When the SONIC acquires the bus, it only transfers data to/from a single area in memory (i.e., TDA, TBA, RDA, RBA, RRA, or CDA). Thus, the bus status pins remain stable for the duration of the block transfer cycle with the following three exceptions: 1) If the SONIC is accessed during a block transfer, S2-S0 indicates bus idle during the register access, then returns to the previous status. 2) If the SONIC finishes writing the Source Address during a block transfer S2-S0 changes from [0,1,0] to [0,1,1]. 3) During an RDA access between the RXpkt.seq\_no and RXpkt.link access, and between the RXpkt.link and RXpkt.in\_use access, S2-S0 will respectively indicate idle [1,1,1] for 2 or 1 bus clocks. Status will be valid on the falling edge of  $\overline{AS}$  or rising edge of  $\overline{ADS}$ .

Figure 5-7 illustrates the SONIC's transitions through memory during the process of transmission and reception. During transmission, the SONIC reads the descriptor information from the TDA and then transmits data of the packet from the TBA. The SONIC moves back and forth between the TDA and TBA until all fragments and packets are transmitted. During reception, the SONIC takes one of two paths. In the first case (path A), when the SONIC detects EOL=0 from the previous reception, it buffers the accepted packet into the RBA, and then writes the descriptor information to the RDA. If the RBA becomes depleted (i.e., RBWC0,1 < EOBC), it moves to the RRA to read a resource descriptor. In the second case (path B), when the SONIC detects EOL=1 from the previous reception, it rereads the

RXpkt.link field to determine if the system has reset the EOL bit since the last reception. If it has, the SONIC buffers the packet as in the first case. Otherwise, it rejects the packet and returns to idle.

### 5.4.4 Bus Mode Compatibility

For compatibility with different microprocessor and bus architectures, the SONIC operates in one of two modes (set by the BMODE pin) called the National/Intel or little endian mode (BMODE tied low) and the Motorola or big endian mode (BMODE tied high). The definitions for several pins change depending on the mode the SONIC is in. Table 5-3 shows these changes. These modes affect both master and slave bus operations with the SONIC.

TABLE 5-3. Bus Mode Compatibility

Pin Name	BMODE = 0 (National/Intel)	BMODE = 1 (Motorola)
$\overline{BR}/\text{HOLD}$	HOLD	$\overline{BR}$
$\overline{BG}/\text{HLDA}$	HLDA	$\overline{BG}$
$\overline{MRW}/\overline{MWR}$	$\overline{MWR}$	$\overline{MRW}$
$\overline{SRW}/\overline{SWR}$	$\overline{SWR}$	$\overline{SRW}$
$\overline{DSACK0}/\overline{RDYi}$	$\overline{RDYi}$	$\overline{DSACK0}$
$\overline{DSACK1}/\overline{RDYo}$	$\overline{RDYo}$	$\overline{DSACK1}$
$\overline{AS}/\overline{ADS}$	$\overline{ADS}$	$\overline{AS}$
$\overline{INT}/\text{INT}$	INT	$\overline{INT}$

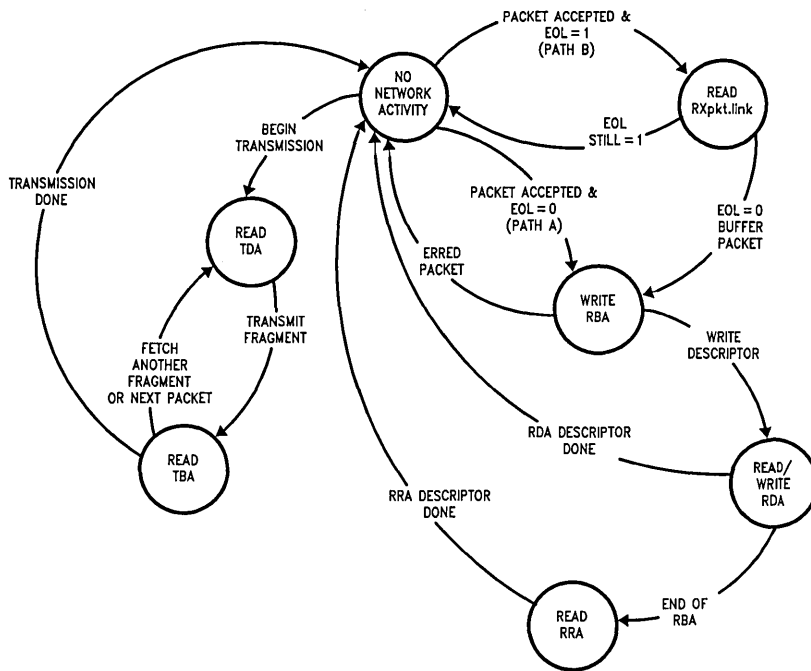


FIGURE 5-7. Bus Status Transitions

TL/F/10492-29

## 5.0 Bus Interface (Continued)

### 5.4.5 Master Mode Bus Cycles

In order to add additional compatibility with different bus architectures, there are two other modes that affect the operation of the bus. These modes are called the synchronous and asynchronous modes and are programmed by setting or resetting the SBUS bit in the Data Configuration Register (DCR). The synchronous and asynchronous modes do not have an effect on slave accesses to the SONIC but they do affect the master mode operation. Within the particular bus/processor mode, synchronous and asynchronous modes are very similar. This section discusses all four modes of operation of the SONIC (National/Intel vs. Motorola, synchronous vs. asynchronous) when it is a bus master.

In this section, the rising edge of T1 and T2 means the beginning of these states, and the falling edge of T1 and T2 means the middle of these states.

#### 5.4.5.1 Adding Wait States

To accommodate different memory speeds, the SONIC provides two methods for adding wait states for its bus operations. Both of these methods can be used individually or in conjunction with each other. A memory cycle is extended by adding additional T2 states. The first method inserts wait-states by withholding the assertion of  $\overline{DSACK0,1}$ / $\overline{STERM}$  or  $\overline{RDYi}$ . The other method allows software to program wait-states. Programming the WC0, WC1 bits in the Data Configuration Register allows 1 to 3 wait-states to be added on each memory cycle. These wait states are inserted between the T1 and T2 bus states and are called T2(wait) bus states. The SONIC will not look at the  $\overline{DSACK0,1}$ ,  $\overline{STERM}$  or  $\overline{RDYi}$  lines until the programmed wait states have passed. Hence,

in order to complete a bus operation that includes programmed wait states, the  $\overline{DSACK0,1}$ ,  $\overline{STERM}$  or  $\overline{RDYi}$  lines must be asserted at their proper times at the end of the cycle during the last T2, not during a programmed wait state. The only exception to this is asynchronous mode where  $\overline{DSACK0,1}$  or  $\overline{RDYi}$  would be asserted during the last programmed wait state, T2 (wait). See the timing for these signals in the timing diagrams for more specific information. Programmed wait states do not affect Slave Mode bus cycles.

#### 5.4.5.2 Memory Cycle for BMODE = 1, Synchronous Mode

On the rising edge of T1, the SONIC asserts  $\overline{ECS}$  to indicate that the memory cycle is starting. The address (A31-A1), bus status (S2-S0) and the direction strobe (MRW) are driven and do not change for the remainder of the memory cycle. On the falling edge of T1, the SONIC deasserts  $\overline{ECS}$  and asserts  $\overline{AS}$ .

In synchronous mode,  $\overline{DSACK0,1}$  are sampled on the rising edge of T2. T2 states will be repeated until  $\overline{DSACK0,1}$  are sampled properly in a low state.  $\overline{DSACK0,1}$  must meet the setup and hold times with respect to the rising edge of bus clock for proper operation.

During read cycles (*Figure 5-8*) data (D31-D0) is latched at the falling edge of T2 and  $\overline{DS}$  is asserted at the falling edge of T1. For write cycles (*Figure 5-9*) data is driven on the rising edge of T1. If there are wait states inserted,  $\overline{DS}$  is asserted on the falling edge of T2.  $\overline{DS}$  is not asserted for zero wait state write cycles. The SONIC terminates the memory cycle by deasserting  $\overline{AS}$  and  $\overline{DS}$  at the falling edge of T2.

## 5.0 Bus Interface (Continued)

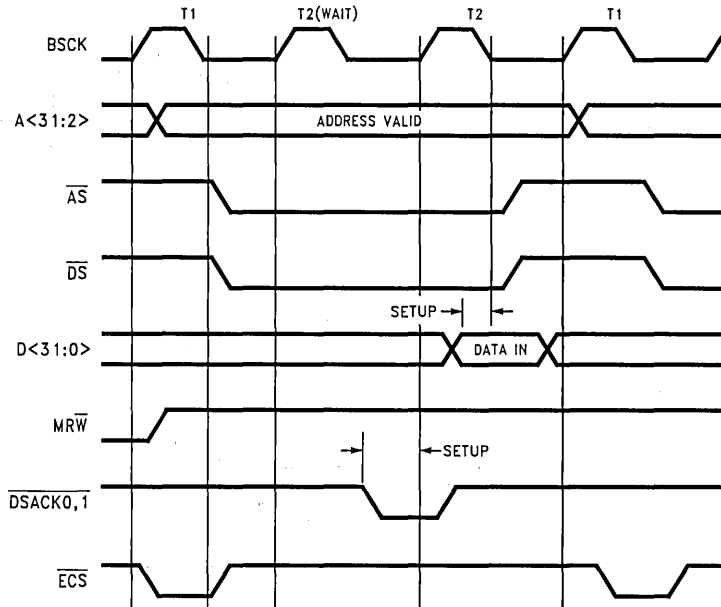


FIGURE 5-8. Memory Read, BMODE = 1, Synchronous (1 Wait-State)

TL/F/10492-31

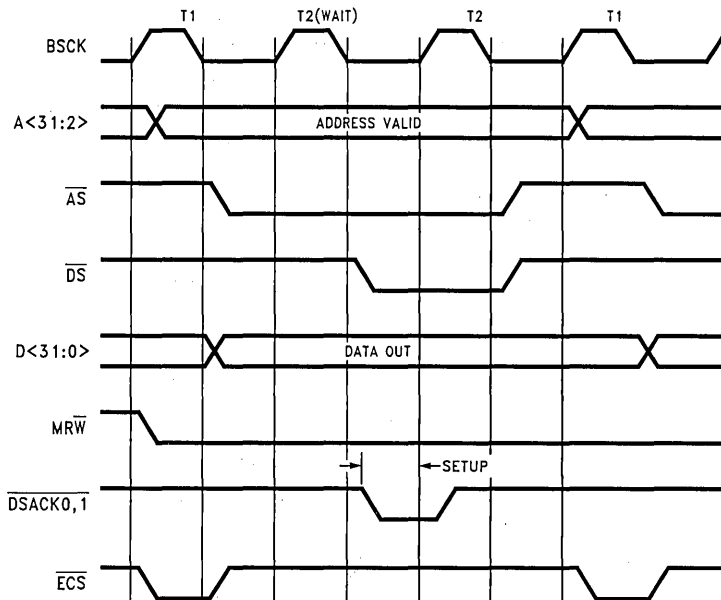


FIGURE 5-9. Memory Write, BMODE = 1, Synchronous (1 Wait-State)

TL/F/10492-33

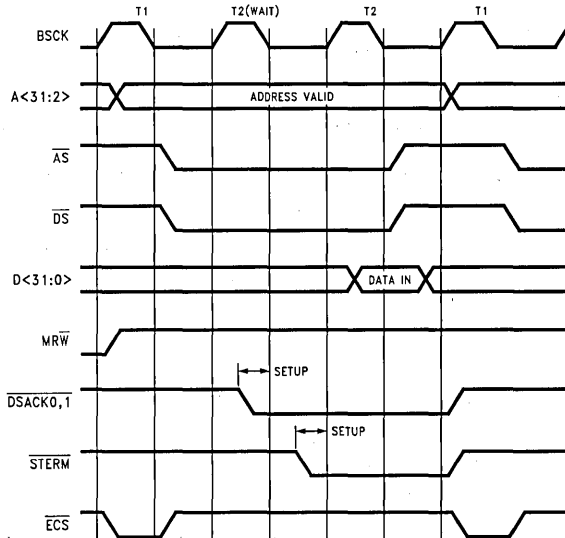
## 5.0 Bus Interface (Continued)

### 5.4.5.3 Memory Cycle for BMODE = 1, Asynchronous Mode

On the rising edge of T1, the SONIC asserts  $\overline{ECS}$  to indicate that the memory cycle is starting. The address (A31-A1), bus status (S2-S0) and the direction strobe (MRW) are driven and do not change for the remainder of the memory cycle. On the falling edge of T1, the SONIC deasserts  $\overline{ECS}$  and asserts  $\overline{AS}$ .

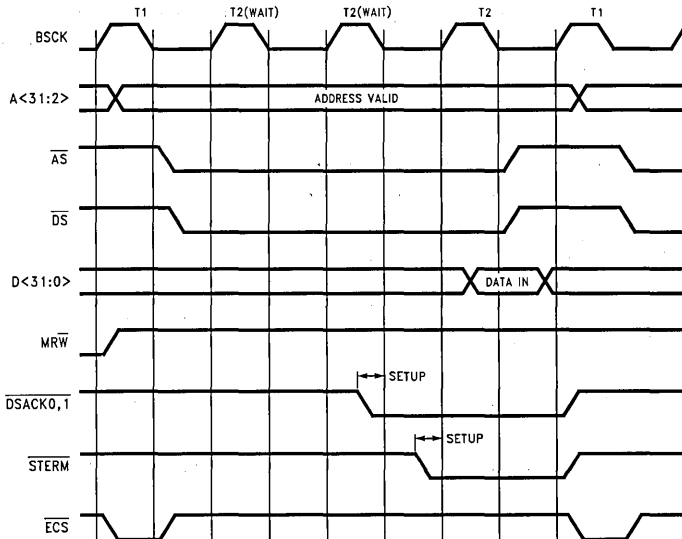
In asynchronous mode,  $\overline{DSACK0,1}$  are asynchronously sampled on the falling edge of both T1 and T2.  $\overline{DSACK0,1}$

do not need to be synchronized to the bus clock because the chip always resolves these signals to either a high or low state. If a synchronous termination of the bus cycle is required, however,  $\overline{STERM}$  may be used.  $\overline{STERM}$  is sampled on the rising edge of T2 and must meet the setup and hold times with respect to that edge for proper operation. Meeting the setup time for  $\overline{DSACK0,1}$  or  $\overline{STERM}$  guarantees that the SONIC will terminate the memory cycle 1.5



TL/F/10492-36

FIGURE 5-10. Memory Read, BMODE = 1, Asynchronous (1 Wait-State)



TL/F/10492-37

FIGURE 5-11. Memory Read, BMODE = 1, Asynchronous (2 Wait-State)

## 5.0 Bus Interface (Continued)

bus clocks after  $\overline{DSACK0,1}$  were sampled, or 1 cycle after  $\overline{STERM}$  was sampled. T2 states will be repeated until  $\overline{DSACK0,1}$  or  $\overline{STERM}$  are sampled properly in a low state (see note below).

During read cycles (Figures 5-10 and 5-11), data (D31:0) is latched at the falling edge of T2 and  $\overline{DS}$  is asserted at the falling edge of T1. For write cycles (Figures 5-12 and 5-13) data is driven on the rising edge of T1. If there are wait

states inserted,  $\overline{DS}$  is asserted on the falling edge of the first T2 (wait).  $\overline{DS}$  is not asserted for zero wait state write cycles. The  $\overline{SONIC}$  terminates the memory cycle by deasserting  $\overline{AS}$  and  $\overline{DS}$  at the falling edge of T2.

**Note:** If the setup time for  $\overline{DSACK0,1}$  is met during T1, or the setup time for  $\overline{STERM}$  is met during the first T2, the full asynchronous bus cycle will take only 2 bus clocks. This may be an unwanted situation. If so,  $\overline{DSACK0,1}$  and  $\overline{STERM}$  should normally be deasserted during T1 and the start of T2 respectively.

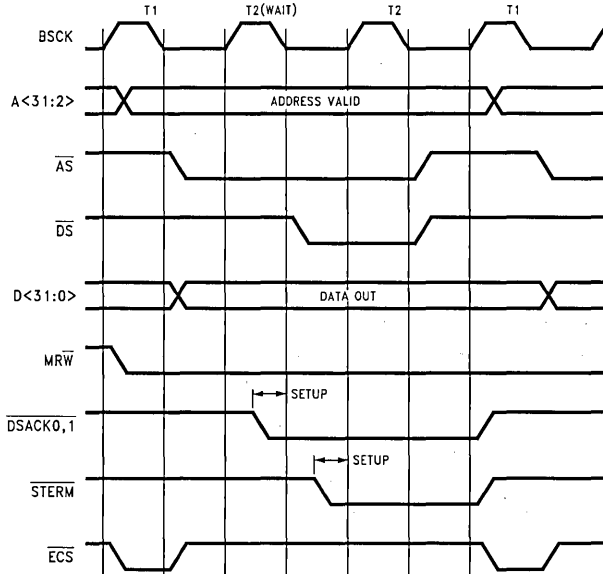


FIGURE 5-12. Memory Write, BMODE = 1, Asynchronous (1 Wait-State)

TL/F/10492-34

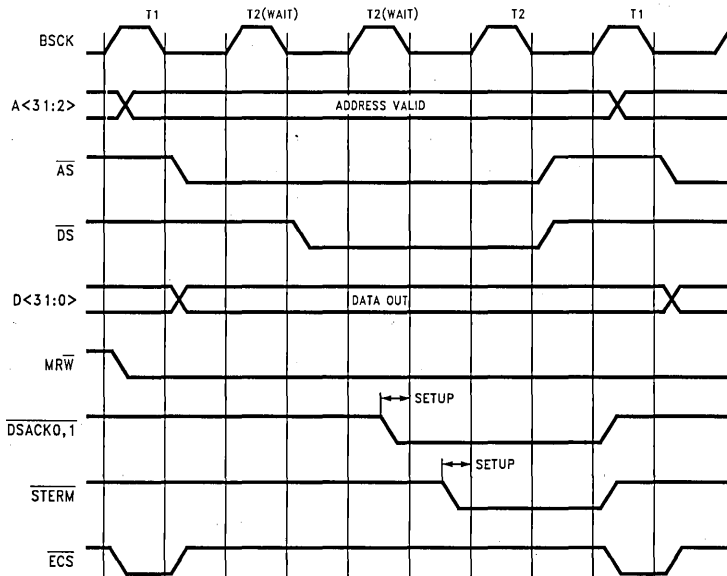


FIGURE 5-13. Memory Write, BMODE = 1, Asynchronous (2 Wait-State)

TL/F/10492-35

## 5.0 Bus Interface (Continued)

### 5.4.5.4 Memory Cycle for BMODE = 0, Synchronous Mode

On the rising edge of T1, the SONIC asserts  $\overline{ADS}$  and  $\overline{ECS}$  to indicate that the memory cycle is starting. The address (A31-A1), bus status (S2-S0) and the direction strobe ( $\overline{MWR}$ ) are driven and do not change for the remainder of the memory cycle. On the falling edge of T1, the SONIC deasserts  $\overline{ECS}$ .  $\overline{ADS}$  is deasserted on the rising edge of T2.

In Synchronous mode,  $\overline{RDYi}$  is sampled on the rising edge at the end of T2 (the rising edge of the next T1). T2 states will be repeated until  $\overline{RDYi}$  is sampled properly in a low state.  $\overline{RDYi}$  must meet the setup and hold times with respect to the rising edge of bus clock for proper operation. During read cycles (Figure 5-14), data (D31-D0) is latched at the rising edge at the end of T2. For write cycles (Figure 5-15) data is driven on the rising edge of T1 and stays driven until the end of the cycle.

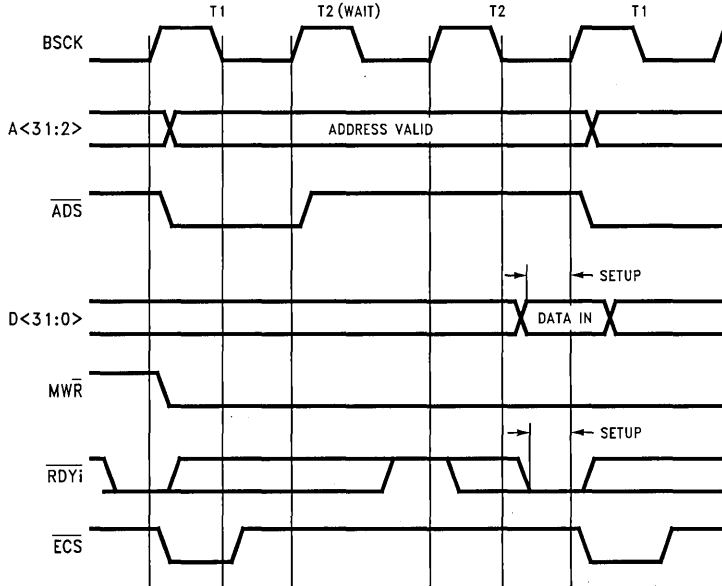


FIGURE 5-14. Memory Read, BMODE=0, Synchronous (1 Wait-State)

TL/F/10492-38

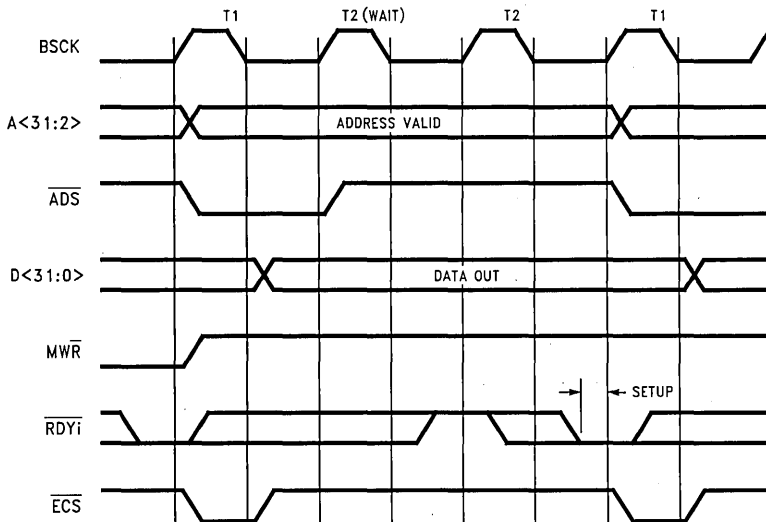


FIGURE 5-15. Memory Write, BMODE=0, Synchronous (1 Wait-State)

TL/F/10492-40

## 5.0 Bus Interface (Continued)

### 5.4.5.5 Memory Cycle for BMODE = 0, Asynchronous Mode

On the rising edge of T1, the SONIC asserts  $\overline{ADS}$  and  $\overline{ECS}$  to indicate that the memory cycle is starting. The address (A31-A1), bus status (S2-S0) and the direction strobe ( $\overline{MWR}$ ) are driven and do not change for the remainder of the memory cycle. On the falling edge of T1, the SONIC deasserts  $\overline{ECS}$ .  $\overline{ADS}$  is deasserted on the rising edge of T2.

In Asynchronous mode,  $\overline{RDYi}$  is asynchronously sampled on the falling edge of both T1 and T2.  $\overline{RDYi}$  does not need to be synchronized to the bus clock because the chip always resolves these signals to either a high or low state. Meeting the setup time for  $\overline{RDYi}$  guarantees that the SONIC will terminate the memory cycle 1.5 bus clocks after  $\overline{RDYi}$  was sampled. T2 states will be repeated until  $\overline{RDYi}$  is sampled properly in a low state (see note on following page).

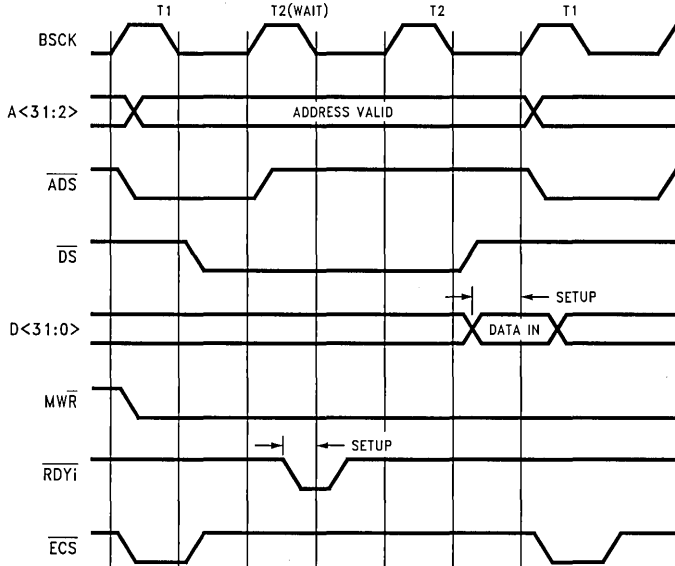


FIGURE 5-16. Memory Read, BMODE = 0, Asynchronous (1 Wait-State)

TL/F/10492-44

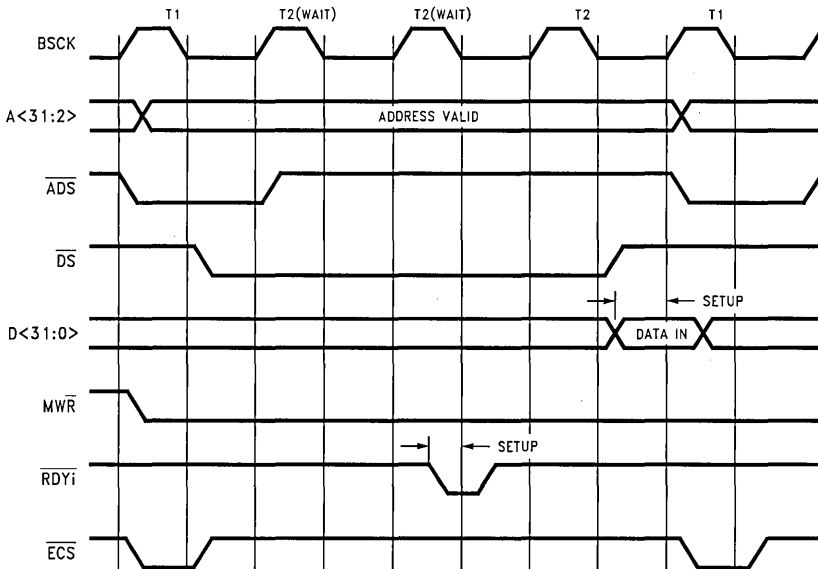


FIGURE 5-17. Memory Read, BMODE = 0, Asynchronous (2 Wait-State)

TL/F/10492-45

## 5.0 Bus Interface (Continued)

During read cycles (*Figures 5-16 and 5-17*), data (D31-D0) is latched on the rising edge at the end of T2 and  $\overline{DS}$  is asserted at the falling edge of T1. For write cycles (*Figures 5-18 and 5-19*) data is driven on the rising edge of T1. If there are wait states inserted,  $\overline{DS}$  is asserted on the falling edge of the first T2(wait).  $\overline{DS}$  is not asserted for zero wait state write cycles. The SONIC terminates the memory cycle by deasserting  $\overline{DS}$  at the falling edge of T2.

**Note:** If the setup time for  $\overline{RDYi}$  is met during T1, the full asynchronous bus cycle will take only 2 bus clocks. This may be an unwanted situation. If so,  $\overline{RDYi}$  should be deasserted during T1.

### 5.4.6 Bus Exceptions (Bus Retry)

The SONIC provides the capability of handling errors during the execution of the bus cycle (*Figure 5-20*).

The system asserts  $\overline{BRT}$  (bus retry) to force the SONIC to repeat the current memory cycle. When the SONIC detects the assertion of  $\overline{BRT}$ , it completes the memory cycle at the end of T2 and gets off the bus by deasserting  $\overline{BGACK}$  or HOLD. Then, if Latched Bus Retry mode is not set (LBR in the Data Configuration Register, Section 4.3.2), the SONIC requests the bus again to retry the same memory cycle. If

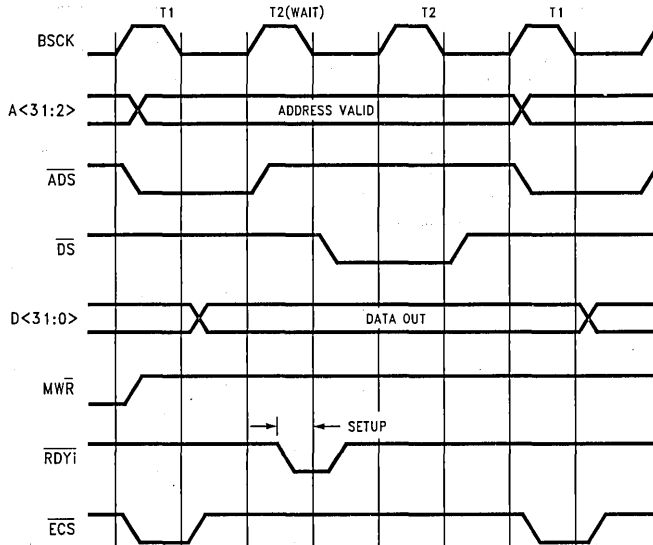


FIGURE 5-18. Memory Write, BMODE=0, Asynchronous (1 Wait-State)

TL/F/10492-42

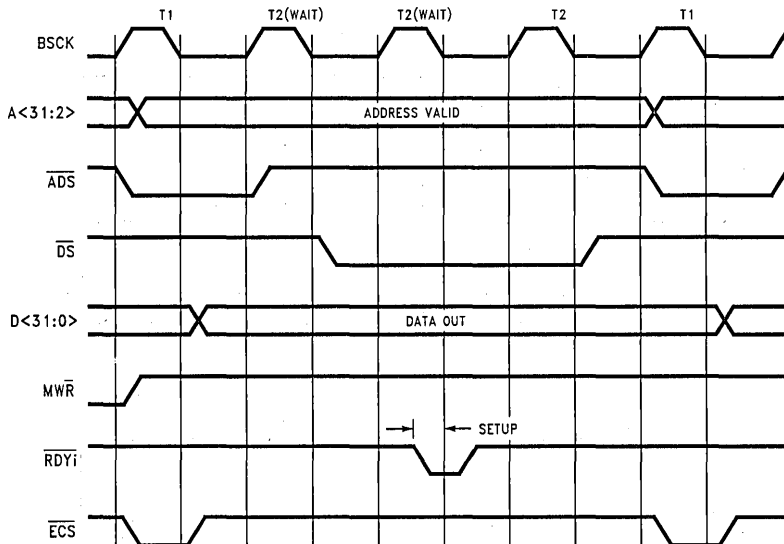


FIGURE 5-19. Memory Write, BMODE=0, Asynchronous (2 Wait-State)

TL/F/10492-43



## 5.0 Bus Interface (Continued)

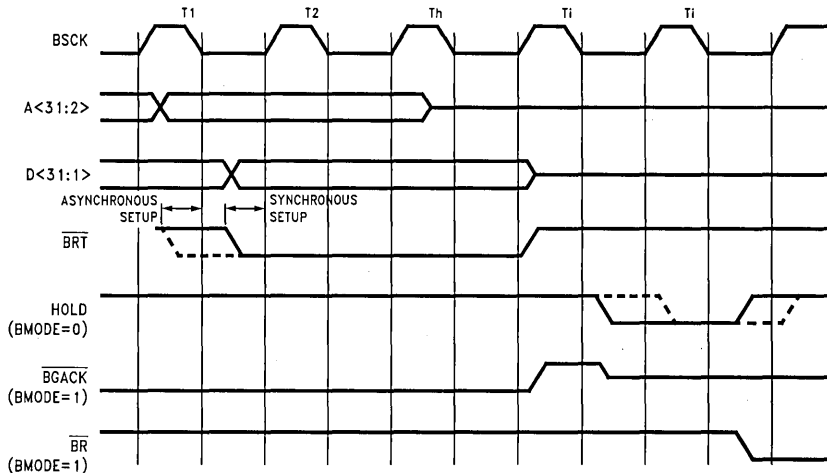


FIGURE 5-20. Bus Exception (Bus Retry)

TL/F/10492-46

Latched Bus Retry is set though, the SONIC will not retry until the BR bit in the ISR (see Section 4.3.6) has been reset and  $\overline{\text{BRT}}$  is deasserted.  $\overline{\text{BRT}}$  has precedence of terminating a memory cycle over  $\overline{\text{DSACK0,1}}$ ,  $\overline{\text{STERM}}$  or  $\overline{\text{RDYi}}$ .

$\overline{\text{BRT}}$  may be sampled synchronously or asynchronously by setting the EXBUS bit in the DCR (see Section 4.3.2). If synchronous Bus Retry is set,  $\overline{\text{BRT}}$  is sampled on the rising edge of T2. If asynchronous Bus Retry is set,  $\overline{\text{BRT}}$  is double synchronized from the falling edge of T1. The asynchronous setup time does not need to be met, but doing so will guarantee that the bus exception will occur in the current bus cycle instead of the next bus cycle. Asynchronous Bus Retry may only be used when the SONIC is set to asynchronous mode.

**Note 1:** The deassertion edge of HOLD is dependent on the PH bit in the DCR2 (see Section 4.3.7). Also,  $\overline{\text{BGACK}}$  is driven high for about 1/2 bus clock before going TRI-STATE.

**Note 2:** If Latched Bus Retry is set,  $\overline{\text{BRT}}$  need only satisfy its setup time (the hold time is not important). Otherwise,  $\overline{\text{BRT}}$  must remain asserted until after the Th state.

**Note 3:** If  $\overline{\text{DSACK0,1}}$ ,  $\overline{\text{STERM}}$  or  $\overline{\text{RDYi}}$  remain asserted after  $\overline{\text{BRT}}$ , the next memory cycle may be adversely affected.

### 5.4.7 Slave Mode Bus Cycle

The SONIC's internal registers can be accessed by one of two methods (BMODE = 1 or BMODE = 0). In both methods, the SONIC is a slave on the bus. This section describes the SONIC's slave mode bus operations.

#### 5.4.7.1 Slave Cycle for BMODE = 1

The system accesses the SONIC by driving  $\overline{\text{SAS}}$ ,  $\overline{\text{CS}}$ ,  $\overline{\text{SRW}}$  and RA<5:0>. SONIC will start a slave cycle once  $\overline{\text{CS}}$  and  $\overline{\text{SAS}}$  are asserted properly. SONIC samples  $\overline{\text{CS}}$  asynchronously at the falling edge of each BSCCK.  $\overline{\text{SAS}}$  signal can be asserted anytime as long as it is before the next falling edge of the clock that the  $\overline{\text{CS}}$  is sampled on.

The register address RA<5:0> and the read/write signal  $\overline{\text{SRW}}$  will be latched by the SONIC on the falling edge of the  $\overline{\text{SAS}}$  signal. Once  $\overline{\text{SAS}}$  and  $\overline{\text{CS}}$  are asserted,  $\overline{\text{SMACK}}$  will be asserted by the SONIC to signify that the SONIC has started the slave cycle. Although  $\overline{\text{CS}}$  and  $\overline{\text{SAS}}$  are asynchronous inputs, meeting their setup times (as shown in Figures 5-21 and 5-22) will guarantee that  $\overline{\text{SMACK}}$ , which is asserted off

of a falling edge, will be asserted 1 bus clock after the falling edge that  $\overline{\text{CS}}$  was clocked in on. This is assuming that the SONIC is not a bus master when  $\overline{\text{CS}}$  was asserted. If the SONIC is a bus master, then, when  $\overline{\text{CS}}$  is asserted, the SONIC will complete its current master bus cycle and get off the bus temporarily (see Section 5.4.8). In this case,  $\overline{\text{SMACK}}$  will be asserted maximum 5 bus clocks after the falling edge that  $\overline{\text{CS}}$  was clocked in on. This is assuming that there were no wait states in the current master mode access. Wait states will increase the time for  $\overline{\text{SMACK}}$  to go low by the number of wait states in the cycle.

If the slave access is a read cycle (Figure 5-21), then the data will be driven off the same edge as  $\overline{\text{SMACK}}$ . If it is a write cycle (Figure 5-22), then the data will be latched in exactly 2 bus clocks after the assertion of  $\overline{\text{SMACK}}$ . In either case,  $\overline{\text{DSACK0,1}}$  are driven low 2 bus clocks after  $\overline{\text{SMACK}}$  to terminate the slave cycle. For a read cycle, the assertion of  $\overline{\text{DSACK0,1}}$  indicates valid register data and for a write cycle, the assertion indicates that the SONIC has latched the data. The SONIC deasserts  $\overline{\text{DSACK0,1}}$  at the rising edge of  $\overline{\text{SAS}}$  or  $\overline{\text{CS}}$  depending on which is deasserted first. The data bus is deasserted on the rising edge of  $\overline{\text{SAS}}$ . The SONIC deasserts  $\overline{\text{SMACK}}$  and causes  $\overline{\text{DSACK0,1}}$  to become TRI-STATE on the falling edge of the BSCCK that  $\overline{\text{SAS}}$  was sampled high on.

**Note 1:** Although the SONIC responds as a 32-bit peripheral when it drives both  $\overline{\text{DSACK0}}$  and  $\overline{\text{DSACK1}}$  low, it transfers data only on lines D<15:0>.

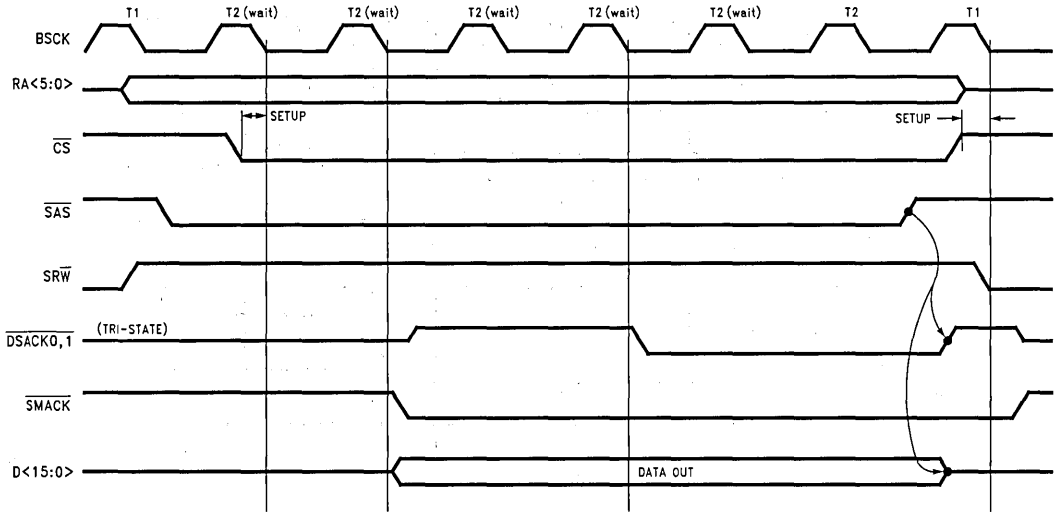
**Note 2:** For multiple register accesses,  $\overline{\text{CS}}$  can be held low and  $\overline{\text{SAS}}$  can be used to delimit the slave cycle. In this case,  $\overline{\text{SMACK}}$  will be driven low due to  $\overline{\text{SAS}}$  going low since  $\overline{\text{CS}}$  has already been asserted. Notice that this means  $\overline{\text{SMACK}}$  will not stay asserted low during the entire time  $\overline{\text{CS}}$  is low (as is the case for  $\overline{\text{MREQ}}$ , Section 5.4.8).

**Note 3:** If memory request ( $\overline{\text{MREQ}}$ ) follows a chip select ( $\overline{\text{CS}}$ ), it must be asserted at least 2 bus clocks after  $\overline{\text{CS}}$  is deasserted. Both  $\overline{\text{CS}}$  and  $\overline{\text{MREQ}}$  must not be asserted concurrently.

**Note 4:** When  $\overline{\text{CS}}$  is deasserted, it must remain deasserted for at least one bus clock.

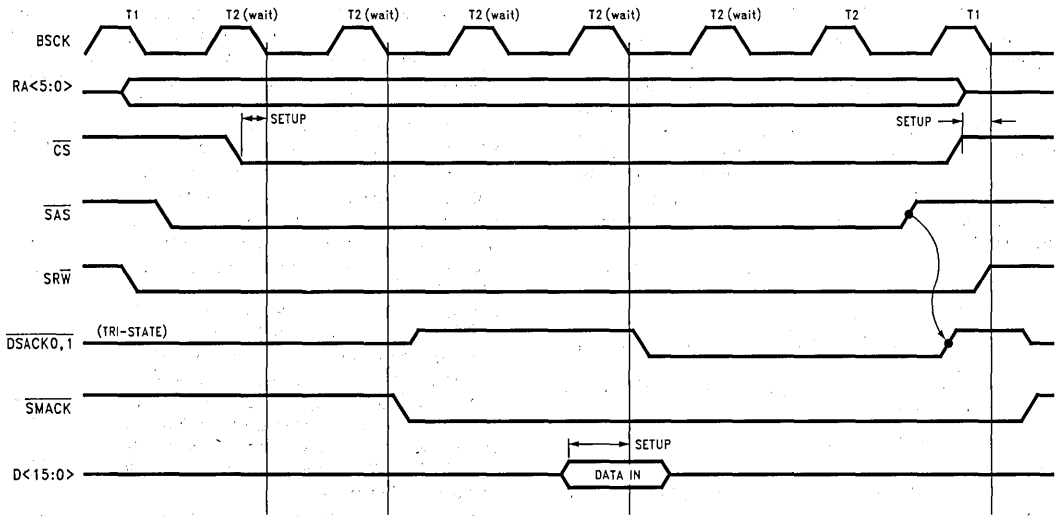
**Note 5:** The way in which  $\overline{\text{SMACK}}$  is asserted due to  $\overline{\text{CS}}$  is not the same as the way in which  $\overline{\text{SMACK}}$  is asserted due to  $\overline{\text{MREQ}}$ . The assertion of  $\overline{\text{SMACK}}$  is dependent upon both  $\overline{\text{CS}}$  and  $\overline{\text{SAS}}$  being low, not just  $\overline{\text{CS}}$ . This is not the same as the case for  $\overline{\text{MREQ}}$  (see Section 5.4.8). The assertion of  $\overline{\text{SMACK}}$  in these two cases should not be confused.

### 5.0 Bus Interface (Continued)



TL/F/10492-47

FIGURE 5-21. Register Read, BMODE = 1



TL/F/10492-48

FIGURE 5-22. Register Write, BMODE = 1

## 5.0 Bus Interface (Continued)

### 5.4.7.2 Slave Cycle for BMODE = 0

The system accesses the SONIC by driving  $\overline{SAS}$ ,  $\overline{CS}$ ,  $\overline{SWR}$  and  $RA <5:0>$ . SONIC will start a slave cycle once  $\overline{CS}$  and  $\overline{SAS}$  are asserted properly. SONIC samples  $\overline{CS}$  asynchronously at the falling edge of each BCLK.  $\overline{SAS}$  signal may be asserted low anytime before or simultaneously to the falling edge of the  $\overline{CS}$  and the deassertion of  $\overline{SAS}$  will start the slave cycle.  $\overline{CS}$  should not be asserted low before the falling edge of  $\overline{SAS}$  as this will cause improper slave operation. The register address  $RA <5:0>$  and the read/write signal  $\overline{SWR}$  will be latched by the SONIC on the rising edge of the  $\overline{SAS}$  signal. Once  $\overline{CS}$  is asserted and  $\overline{SAS}$  is deasserted,  $\overline{SMACK}$  will be asserted by the SONIC to signify that the SONIC has started the slave cycle. Although  $\overline{CS}$  and  $\overline{SAS}$  are asynchronous inputs, meeting their setup times (as shown in Figures 5-23 and 5-24) will guarantee that  $\overline{SMACK}$ , which is asserted off a falling edge, will be asserted on the falling edge of the BCLK and  $\overline{SAS}$  was sampled high on. This is assuming that the SONIC is not a bus master when  $\overline{CS}$  is asserted. If the SONIC is a bus master, then, when  $\overline{CS}$  is asserted, the SONIC will complete its current master bus cycle and get off the bus temporarily (see Section 5.4.8). In this case,  $\overline{SMACK}$  will be asserted maximum 4 bus clocks after the falling edge of BCLK that  $\overline{SAS}$  was sampled high on. This is assuming that there were no wait states in the current master mode access. Wait states will increase the time for  $\overline{SMACK}$  to go low by the number of wait states in the cycle.

If the slave access is a read cycle (Figure 5-23), then the data will be driven off the same edge as  $\overline{SMACK}$ . If it is a write cycle (Figure 5-24), then the data will be latched in exactly 2 bus clocks after the assertion of  $\overline{SMACK}$ . In either case,  $\overline{RDY}_0$  is driven low 2.5 bus clocks after  $\overline{SMACK}$  to terminate the slave cycle. For a read cycle, the assertion of  $\overline{RDY}_0$  indicates valid register data and for a write cycle, the assertion indicates that the SONIC has latched the data. The SONIC deasserts  $\overline{RDY}_0$ ,  $\overline{SMACK}$  and the data if the cycle is a read cycle at the falling edge of  $\overline{SAS}$  or the rising edge of  $\overline{CS}$  depending on which is first.

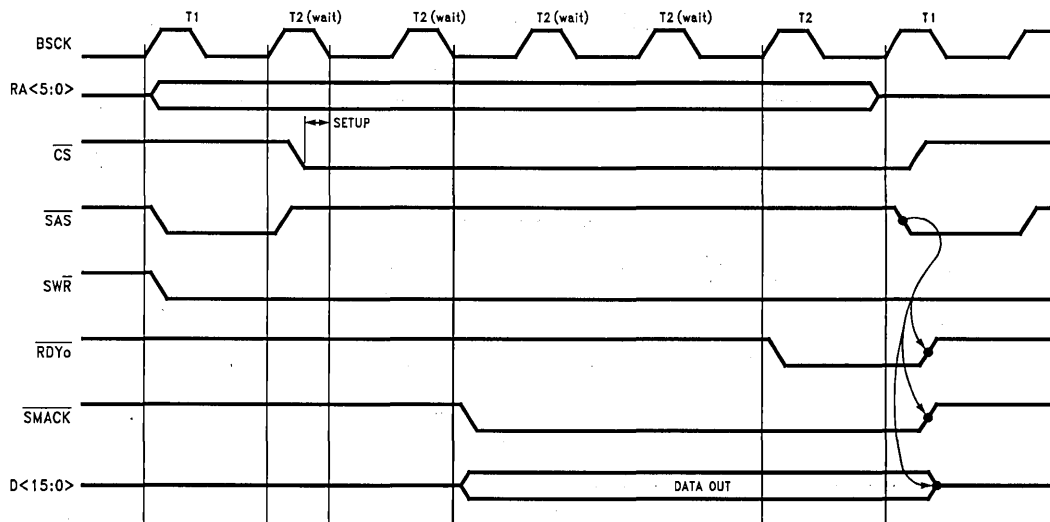
**Note 1:** The SONIC transfers data only on lines  $D <15:0>$  during slave mode accesses.

**Note 2:** For multiple register accesses,  $\overline{CS}$  can be held low and  $\overline{SAS}$  can be used to delimit the slave cycle (this is the only case where  $\overline{CS}$  may be asserted before  $\overline{SAS}$ ). In this case,  $\overline{SMACK}$  will be driven low due to  $\overline{SAS}$  going high since  $\overline{CS}$  has already been asserted. Notice that this means  $\overline{SMACK}$  will not stay asserted low during the entire time  $\overline{CS}$  is low (as is the case for  $\overline{MREQ}$ , see Section 5.4.8).

**Note 3:** If memory request ( $\overline{MREQ}$ ) follows a chip select  $\overline{CS}$ , it must be asserted at least 2 bus clocks after  $\overline{CS}$  is deasserted. Both  $\overline{CS}$  and  $\overline{MREQ}$  must not be asserted concurrently.

**Note 4:** When  $\overline{CS}$  is deasserted, it must remain deasserted for at least one bus clock.

**Note 5:** The way in which  $\overline{SMACK}$  is asserted due to  $\overline{CS}$  is not the same as the way in which  $\overline{SMACK}$  is asserted due to  $\overline{MREQ}$ . The assertion of  $\overline{SMACK}$  is dependent upon both  $\overline{CS}$  and  $\overline{SAS}$  being low, not just  $\overline{CS}$ . This is not the same as the case for  $\overline{MREQ}$  (see Section 5.4.8). The assertion of  $\overline{SMACK}$  in these two cases should not be confused.



TL/F/10492-49

FIGURE 5-23. Register Read, BMODE = 0

### 5.0 Bus Interface (Continued)

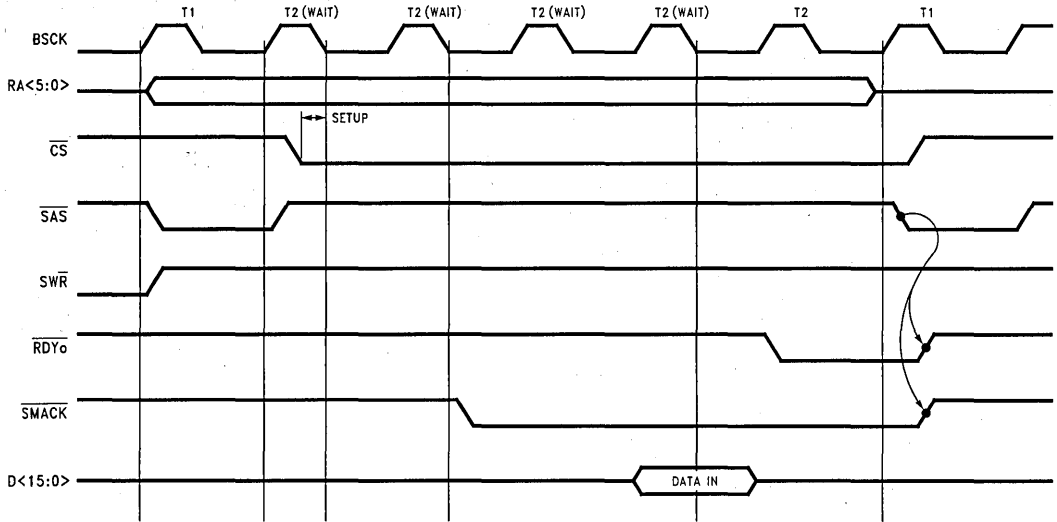


FIGURE 5-24. Register Write, BMODE = 0

TL/F/10492-50

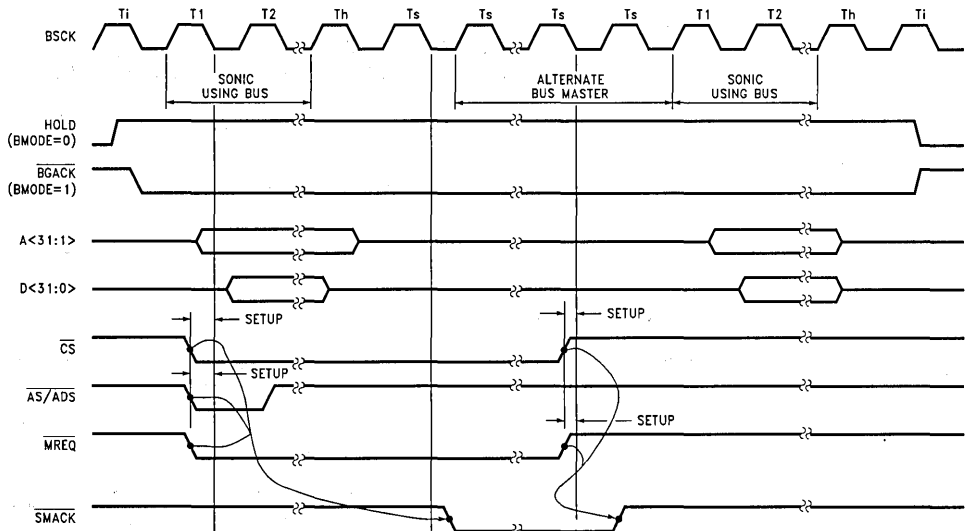


FIGURE 5-25. On-Chip Memory Arbitration

TL/F/10492-51

## 5.0 Bus Interface (Continued)

### 5.4.8 On-Chip Memory Arbiter

For applications which share the buffer memory area with the host system (shared-memory applications), the SONIC provides a fast on-chip memory arbiter for efficiently resolving accesses between the SONIC and the host system (Figure 5-25). The host system indicates its intentions to use the shared-memory by asserting Memory Request (MREQ). The SONIC will allow the host system to use the shared memory by acknowledging the host system's request with Slave and Memory Acknowledge (SMACK). Once SMACK is asserted, the host system may use the shared memory freely. The host system gives up the shared memory by deasserting MREQ.

MREQ is clocked in on the falling edge of bus clock and is double synchronized internally to the rising edge. SMACK is asserted on the falling edge of a Ts bus cycle. If the SONIC is not currently accessing the memory, SMACK is asserted immediately after MREQ was clocked in. If, however, the SONIC is accessing the shared memory, it finishes its current memory transfer and then issues SMACK. SMACK will be asserted one bus clock minimum to five bus clocks maximum after MREQ is clocked in. Since MREQ is double synchronized, it is not necessary to meet its setup time. Meeting the setup time for MREQ will, however, guarantee that SMACK is asserted one to five bus clocks after the current bus clock. SMACK will deassert within one bus clock after MREQ is deasserted. The SONIC will then finish its master operation if it was using the bus previously.

If the host system needs to access the SONIC's registers instead of shared memory, CS would be asserted instead of MREQ. Accessing the SONIC's registers works almost exactly the same as accessing the shared memory except that the SONIC goes into a slave cycle instead of going idle. See Section 5.4.7 for more information about how register accesses work.

**Note 1:** The successive assertion of CS and MREQ must be separated by at least two bus clocks. Both CS and MREQ must not be asserted concurrently.

**Note 2:** The number of bus clocks between MREQ being asserted and the assertion of SMACK when the SONIC is in Master Mode is 5 bus clocks assuming there were no wait states in the Master Mode access. Wait states will increase the time for SMACK to go low by the number of wait states in the cycle (the time will be 5 + the number of wait states).

**Note 3:** The way in which SMACK is asserted due to CS is not the same as the way in which SMACK is asserted due to MREQ. SMACK goes low as a direct result of the assertion of MREQ, whereas, for CS, SAS must also be driven low (BMODE = 1) or high (BMODE = 0) before SMACK will be asserted. This means that when SMACK is asserted due to MREQ, SMACK will remain asserted until MREQ is deasserted. Multiple memory accesses can be made to the shared memory without SMACK ever going high. When SMACK is asserted due to CS, however, SMACK will only remain low as long as SAS is also low (BMODE = 1). SMACK will not remain low throughout multiple register accesses to the SONIC because SAS must toggle for each register access. This is an important difference to consider when designing shared memory designs.

**Note 4:** In Motorola mode, if a bus master uses the MREQ to request the bus from the SONIC, care should be taken to isolate the DSACK0,1 from the bus (e.g., use TRI-STATE buffers) because the DSACK0,1 will be driven by the SONIC even after the SONIC has given up the bus.

### 5.4.9 Chip Reset

The SONIC has two reset modes; a hardware reset and a software reset. The SONIC can be hardware reset by asserting the RESET pin or software reset by setting the RST bit in the Command Register (Section 4.3.1). The two reset modes are not interchangeable since each mode performs a different function.

**TABLE 5-4. Internal Register Content after RESET**

Register	Contents after Reset	
	Hardware Reset	Software Reset
Command	0094h	0094h/00A4h
Data Configuration (DCR and DCR2)	*	unchanged
Interrupt Mask	0000h	unchanged
Interrupt Status	0000h	unchanged
Transmit Control	0101h	unchanged
Receive Control	**	unchanged
End Of Buffer Count	02F8h	unchanged
Sequence Counters	0000h	unchanged
CAM Enable	0000h	unchanged

\*Bits 15 and 13 of the DCR and bits 4 through 0 of the DCR2 are reset to a 0 during a hardware reset. Bits 15-12 of the DCR2 are unknown until written to. All other bits in these two registers are unchanged.

\*\*Bits LB1, LB0 and BRD are reset to a 0 during hardware reset. All other bits are unchanged.

After power-on, the SONIC must be hardware reset before it will become operational. This is done by asserting RESET for a minimum of 10 transmit clocks (10 ethernet transmit clock periods, TXC). If the bus clock (BSCK) period is greater than the transmit clock period, RESET should be asserted for 10 bus clocks instead of 10 transmit clocks. A hardware reset places the SONIC in the following state. (The registers affected are listed in parenthesis. See Table 5-4 and Section 4.3 for more specific information about the registers and how they are affected by a hardware reset. Only those registers listed below and in Table 5-4 are affected by a hardware reset.)

1. Receiver and Transmitter are disabled (CR).
2. The General Purpose timer is halted (CR).
3. All interrupts are masked out (IMR).
4. The NCRS and PTX status bits in the Transmit Control Register (TCR) are set.
5. The End Of Byte Count (EOBC) register is set to 02F8h (760 words).
6. Packet and buffer sequence number counters are set to zero.
7. All CAM entries are disabled. The broadcast address is also disabled (CAM Enable Register and the RCR).
8. Loopback operation is disabled (RCR).
9. The latched bus retry is set to the unlatched mode (DCR).
10. All interrupt status bits are reset (ISR).
11. The Extended Bus Mode is disabled (DCR).
12. HOLD will be asserted/deasserted from the falling clock edge (DCR2).

### 5.0 Bus Interface (Continued)

- 13. Latched Ready Mode is disabled (DCR2).
- 14. PCOMP will not be asserted (DCR2).
- 15. Packets will be accepted (not rejected) on CAM match (DCR2).

A software reset immediately terminates DMA operations and future interrupts. The chip is put into an idle state where registers can be accessed, but the SONIC will not be active in any other way. The registers are affected by a software reset as shown in Table 5-4 (only the Command Register is changed).

### 6.0 Network Interfacing

The SONIC contains an on-chip ENDEC that performs the network interfacing between the AUI (Attachment Unit Interface) and the SONIC's MAC unit. A pin selectable option allows the internal ENDEC to be disabled and the MAC/

ENDEC signals to be supplied to the user for connection to an external ENDEC. If the EXT pin is tied to ground (EXT=0) the internal ENDEC is selected and if EXT is tied to V<sub>CC</sub> (EXT=1) the external ENDEC option is selected.

**Internal ENDEC:** When the internal ENDEC is used (EXT=0) the interface signals between the ENDEC and MAC unit are internally connected. While these signals are used internally by the SONIC they are also provided as an output to the user (Figure 6-1).

The internal ENDEC allows for a 2-chip solution for the complete Ethernet interface. Figure 6-2 shows a typical diagram of the network interface.

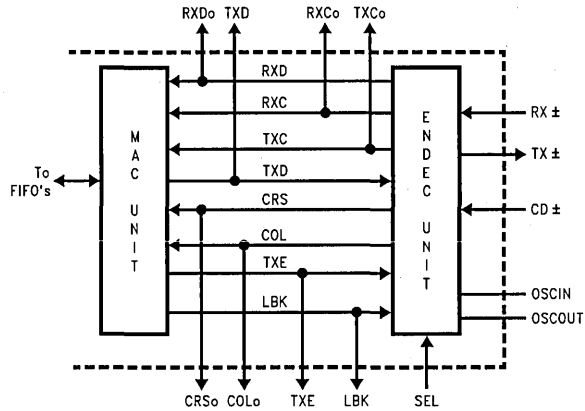
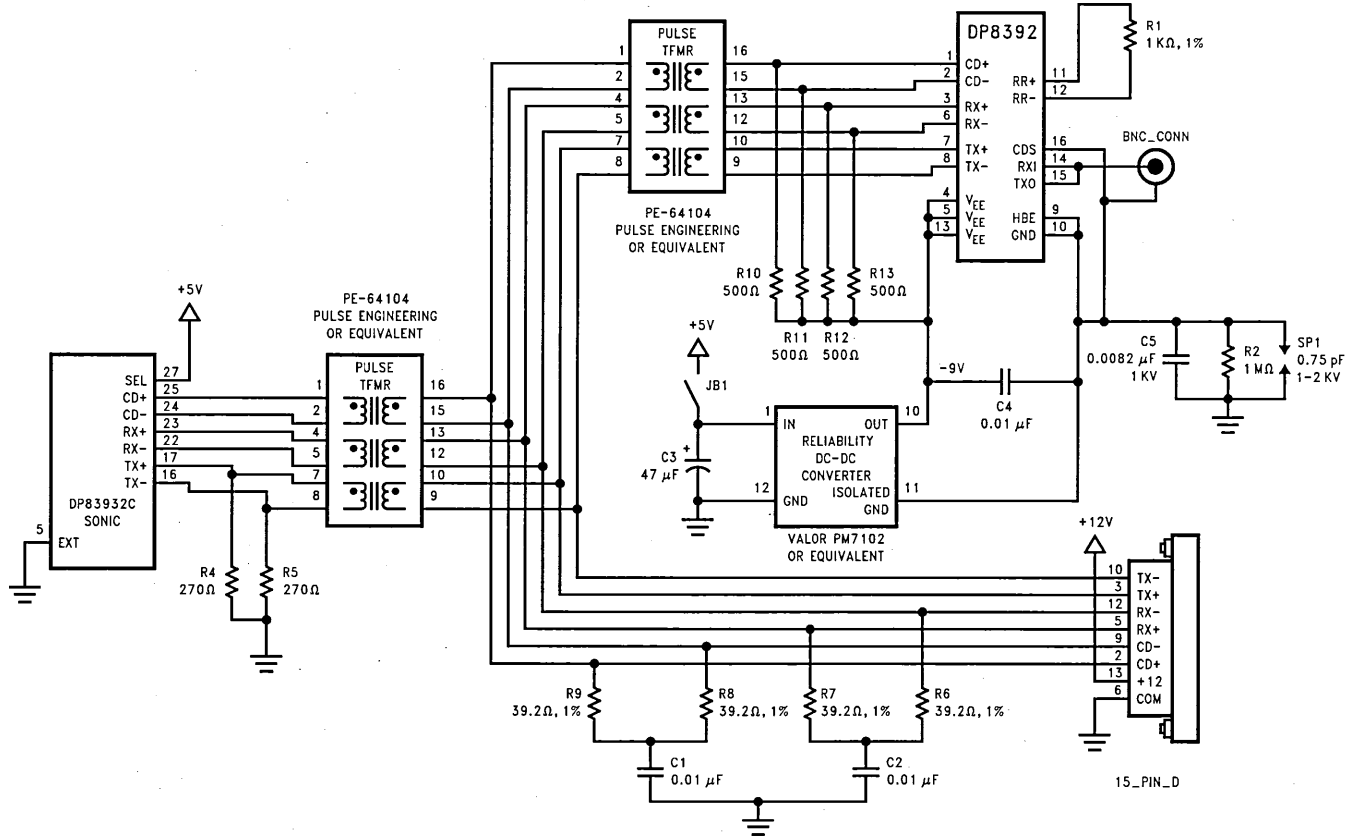


FIGURE 6-1. MAC and Internal ENDEC Interface Signals

TL/F/10492-52



Note: When using BNC-CONN only, R10 to R13 should be 1.5 kΩ each

FIGURE 6-2. Network Interface Example (EXT = 0), using a single jumper, JB1, for network interface selection

TL/F/10492-53

## 6.0 Network Interfacing (Continued)

**External ENDEC:** When EXT = 1 the internal ENDEC is bypassed and the signals are provided directly to the user. Since SONIC's on-chip ENDEC is the same as National's DP83910 Serial Network Interface (SNI) the interface considerations discussed in this section would also apply to using this device in the external ENDEC mode.

### 6.1 MANCHESTER ENCODER AND DIFFERENTIAL DRIVER

The ENDEC unit's encoder begins operation when the MAC section begins sending the serial data stream. It converts NRZ data from the MAC section to Manchester data for the differential drivers (TX+ / -). In Manchester encoding, the first half of the bit cell contains the complementary data and the second half contains the true data (Figure 6-3). A transition always occurs at the middle of the bit cell. As long as the MAC continues sending data, the ENDEC section remains in operation. At the end of transmission, the last transition is always positive, occurring at the center of the bit cell if the last bit is a one, or at the end of the bit cell if the last bit is a zero.

The differential transmit pair drives up to 50 meters of twisted pair AUI cable. These outputs are source followers which require two 270 $\Omega$  pull-down resistors to ground. In addition, a pulse transformer is required between the transmit pair output and the AUI interface.

The driver allows both half-step and full-step modes for compatibility with Ethernet and IEEE 802.3. When the SEL pin is tied to ground (for Ethernet), TX+ is positive with respect to TX- during idle on the primary side of the isolation transformer (Figure 6-2). When SEL is tied to V<sub>CC</sub> (for IEEE 802.3), TX+ and TX- are equal in the idle state.

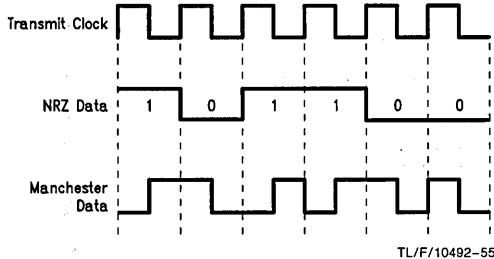


FIGURE 6.3. Manchester Encoded Data Stream

#### 6.1.1 Manchester Decoder

The decoder consists of a differential receiver and a phase lock loop (PLL) to separate the Manchester encoded data stream into clock signals and NRZ data. The differential input must be externally terminated with two 39 $\Omega$  resistors connected in series. In addition, a pulse transformer is required between the receive input pair and the AUI interface. To prevent noise from falsely triggering the decoder, a squelch circuit at the input rejects signals with a magnitude less than -175 mV. Signals more negative than -300 mV are decoded.

Once the input exceeds the squelch requirements, the decoder begins operation. The decoder detects the end of a frame within one and a half bit times after the last bit of data.

#### 6.1.2 Collision Translator

When the Ethernet transceiver (DP8392 CTI) detects a collision, it generates a 10 MHz signal to the differential collision inputs (CD+ and CD-) of the SONIC. When SONIC detects these inputs active, its Collision translator converts the 10 MHz signal to an active collision signal to the MAC section. This signal causes SONIC to abort its current transmission and reschedule another transmission attempt.

The collision differential inputs are terminated the same way as the differential receive inputs and a pulse transformer is required between the collision input pair and the AUI interface. The squelch circuitry is also similar, rejecting pulses with magnitudes less than -175 mV.

#### 6.1.3 Oscillator Inputs

The oscillator inputs to the SONIC (OSCIN and OSCOUT) can be driven with a parallel resonant crystal or an external clock. In either case the oscillator inputs must be driven with a 20 MHz signal. The signal is divided by 2 to generate the 10 MHz transmit clock (TXC) for the MAC unit. The oscillator also provides internal clock signals for the encoding and decoding circuits.

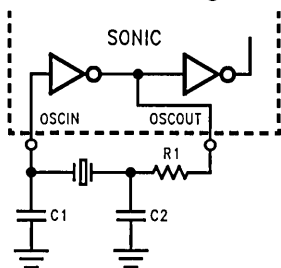
##### 6.1.3.1 External Crystal

According to the IEEE 802.3 standard, the transmit clock (TXC) must be accurate to 0.01%. This means that the oscillator circuit, which includes the crystal and other parts involved must be accurate to 0.01% after the clock has been divided in half. Hence, when using a crystal, it is necessary to consider all aspects of the crystal circuit. An example of a recommended crystal circuit is shown in Figure 6-4 and suggested crystal specifications are shown in Table 6-1.

The load capacitors in Figure 6-4, C1 and C2, should be no greater than 36 pF each, including all stray capacitance (see note 2 below). The resistor, R1, may be required in order to minimize frequency drift due to changes in V<sub>CC</sub>. If R1 is required, its value must be carefully selected since R1 decreases the loop gain. If R1 is made too large, the loop gain will be greatly reduced and the crystal will not oscillate. If R1 is made too small, normal variations in V<sub>CC</sub> may cause the oscillation frequency to drift out of specification. As a first rule of thumb, the value of R1 should be made equal to five times the motional resistance of the crystal. The motional resistance of 20 MHz crystals is usually in the range of 10 $\Omega$  to 30 $\Omega$ . This implies that reasonable values for R1 should be in the range of 50 $\Omega$  to 150 $\Omega$ . The decision of whether or not to include R1 should be based upon measured variations of crystal frequency as each of the circuit parameters are varied.



## 6.0 Network Interfacing (Continued)



TL/F/10492-81

**FIGURE 6.4. Crystal Connection to the SONIC** (see text)

**Note 1:** The OSCOUT pin is not guaranteed to provide a TTL compatible logic output, and should not be used to drive any external logic. If additional logic needs to be driven, then an external oscillator should be used as described in the following section.

**Note 2:** The frequency marked on the crystal is usually measured with a fixed load capacitance specified in the crystal's data sheet. The actual load capacitance used should be the specified value minus the stray capacitance.

**TABLE 6-1. Crystal Specifications**

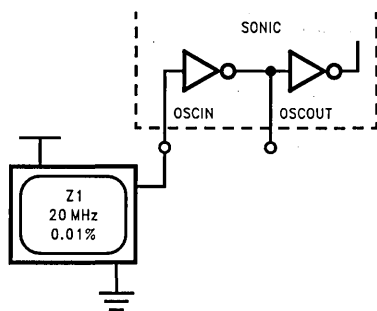
Resonant frequency	20 MHz
Tolerance (see text)	0.01% at 25°C
Accuracy	0.005% (50 ppm) at 0 to 70°C
Fundamental Mode Series Resistance	≤25Ω
Specified Load Capacitance	≤18 pF
Type	AT cut
Circuit	Parallel Resonance

### 6.1.3.2 Clock Oscillator Module

The SONIC also allows for an external clock oscillator to be used. The connection configuration is shown in *Figure 6.5*. This connection requires an oscillator with the following specifications:

1. TTL or CMOS output with a 0.01% frequency tolerance
2. 40%–60% duty cycle (50% duty cycle preferred)
3. One CMOS loads output drive

The above assumes no other circuitry is driven. In this configuration, the OSCOUT pin must be left open.



TL/F/10492-83

**FIGURE 6.5. Oscillator Module Connection to the SONIC**

### 6.1.3.3 PCB Layout Considerations

Care should be taken when connecting a crystal. Stray capacitance (e.g., from PC board traces and plated through holes around the OSCIN and OSCOUT pins) can shift the crystal's frequency out of range, causing the transmitted frequency to exceed the 0.01% tolerance specified by IEEE. The layout considerations for using an external crystal are rather straightforward. The oscillator layout should locate all components close to the OSCIN and OSCOUT pins and should use short traces that avoid excess capacitance and inductance. A solid ground should be used to connect the ground legs of the two capacitors.

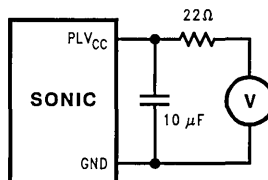
When connecting an external oscillator, the only considerations are to keep the oscillator module as close to the SONIC as possible to reduce stray capacitance and inductance and to give the module a clean V<sub>CC</sub> and a solid ground.

### 6.1.4 Power Supply Considerations

In general, power supply routing and design for the SONIC need only follow standard practices. In some situations, however, additional care may be necessary in the layout of the analog supply. Specifically, special care may be needed for the TXVCC, RXVCC and PLLVCC power supplies and the TXGND and ANGND. In most cases the analog and digital power supplies can be interconnected. However, to ensure optimum performance of the SONIC's analog functions, power supply noise should be minimized. To reduce analog supply noise, any of several techniques can be used.

1. Route analog supplies as a separate set of traces or planes from the digital supplies with their own decoupling capacitors.
2. Provide noise filtering on the analog supply pins by inserting a low pass filter. Alternatively, a ferrite bead could be used to reduce high frequency power supply noise.
3. Utilize a separate regulator to generate the analog supply.

The PLLV<sub>CC</sub> pin is the +5V power supply for the phase lock loop (PLL) of the SONIC ENDEC unit. Since this is an analog circuit, excessive noise on the PLLV<sub>CC</sub> pin can affect the performance of the PLL. This noise, if in the 10 kHz to 400 kHz range, can reduce the jitter performance of the ENDEC, resulting in missing packets or CRC errors. If the power supply noise is causing significant packet reception error, a low pass filter could be added to reduce the power supply noise and hence improve the jitter performance. Standard analog design techniques should be utilized when laying out the power supply traces on the board. If the digital power supply is used, it may be desirable to add a one pole RC filter (designed to have a cut-off frequency of 1 kHz) as shown in *Figure 6.6* to improve the jitter performance. The PLLV<sub>CC</sub> only draws 3 mA–4 mA so the voltage across the resistor is less than 90 mV, which will not affect the PLL's operation.



TL/F/10492-87

**FIGURE 6.6. Filtering the Power Supply Noise**

## 7.0 AC and DC Specifications

### Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage ( $V_{CC}$ )	-0.5V to 7.0V
DC Input Voltage ( $V_{IN}$ )	-0.5V to $V_{CC} + 0.5V$
DC Output Voltage ( $V_{OUT}$ )	-0.5V to $V_{CC} + 0.5V$
Storage Temperature Range ( $T_{STG}$ )	-65°C to 150°C
Power Dissipation (PD)	500 mW
Lead Temp. (TL) (Soldering, 10 sec.)	260°C
ESD Rating	
( $R_{ZAP} = 1.5k, C_{ZAP} = 120 pF$ )	1.5 kV

### DC Specifications $T_A = 0^\circ C$ to $70^\circ C$ , $V_{CC} = 5V \pm 5\%$ , unless otherwise specified

Symbol	Parameter	Conditions	Min	Max	Units
$V_{OH}$	Minimum High Level Output Voltage	$I_{OH} = -8 mA$	3.0		V
$V_{OL}$	Maximum Low Level Output Voltage	$I_{OL} = 8 mA$		0.4	V
$V_{IH}$	Minimum High Level Input Voltage		2.0		V
$V_{IL}$	Maximum Low Level Input Voltage			0.8	V
$I_{IN}$	Input Current	$V_{IN} = V_{CC}$ or GND	-1.0	1.0	$\mu A$
$I_{OZ}$	TRI-STATE Output Leakage Current	$V_{OUT} = V_{CC}$ or GND	-10	10	$\mu A$
$I_{CC}$	Average Operating Supply Current	$I_{OUT} = 0 mA$ , Freq = 20 MHz		90	mA
$I_{CC}$	Average Operating Supply Current	$I_{OUT} = 0 mA$ , Freq = 25 MHz		100	mA
$I_{CC}$	Average Operating Supply Current	$I_{OUT} = 0 mA$ , Freq = 33 MHz		115	mA

#### AUI INTERFACE PINS (TX $\pm$ , RX $\pm$ , and CD $\pm$ )

$V_{OD}$	Diff. Output Voltage (TX $\pm$ )	78 $\Omega$ Termination and 270 $\Omega$ from Each to GND	$\pm 550$	$\pm 1200$	mV
$V_{OB}$	Diff. Output Voltage Imbalance (TX $\pm$ )	78 $\Omega$ Termination and 270 $\Omega$ from Each to GND	Typical: 40 mV		
$V_U$	Undershoot Voltage (TX $\pm$ )	78 $\Omega$ Termination and 270 $\Omega$ from Each to GND	Typical: 80 mV		
$V_{DS}$	Diff. Squelch Threshold (RX $\pm$ and CD $\pm$ )		-175	-300	mV

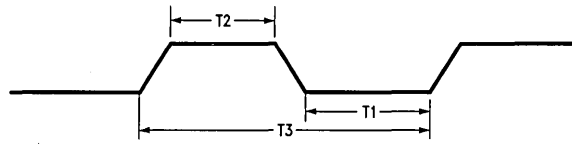
#### OSCILLATOR PINS (OSCIN AND OSCOUT)

$V_{IH}$	OSCIN Input High Voltage	OSCIN is Connected to an Oscillator and OSCOUT is Open	2.0		V
$V_{IL}$	OSCIN Input Low Voltage	OSCIN is Connected to an Oscillator and OSCOUT is Open		0.8	V
$I_{OSC2}$	X2 Input Leakage Current	OSCIN is Connected to an Oscillator and OSCOUT is Open $V_{IN} = V_{CC}$ or GND	-10	10	$\mu A$

## 7.0 AC and DC Specifications (Continued)

### AC Characteristics

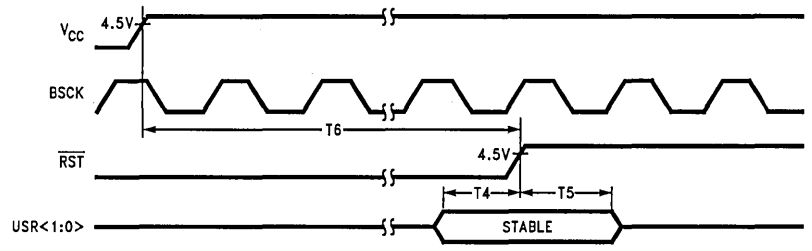
#### BUS CLOCK TIMING



TL/F/10492-56

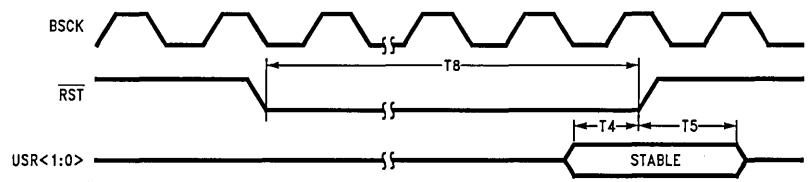
Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T1	Bus Clock Low Time	20		16		13.5		ns
T2	Bus Clock High Time	20		16		13.5		ns
T3	Bus Clock Cycle Time	50		40		30		ns

#### POWER-ON RESET



TL/F/10492-57

#### NON POWER-ON RESET



TL/F/10492-58

Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T4	USR<1:0> Setup to $\overline{\text{RST}}$	7		6		5		ns
T5	USR<1:0> Hold from $\overline{\text{RST}}$	9		8		7		ns
T6	Power-On Reset Low (Notes 1, 2)			10		10		TXC
T8	Reset Pulse Width (Notes 1, 2)			10		10		TXC

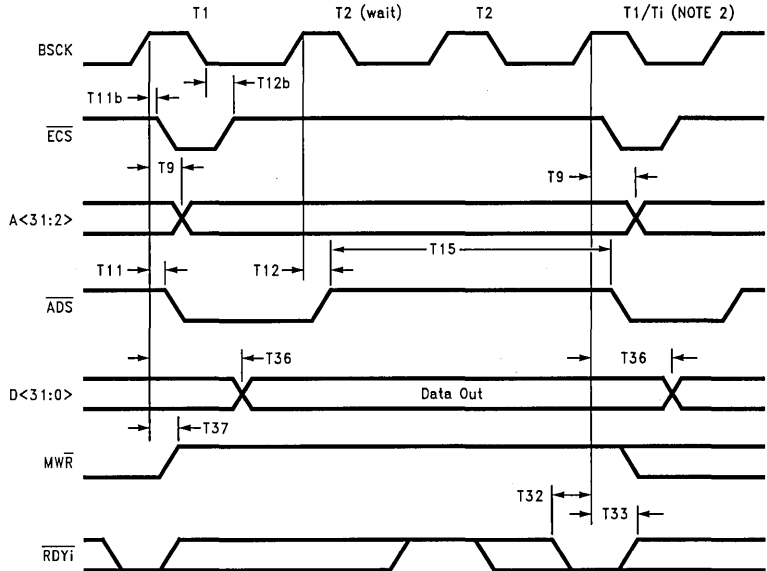
**Note 1:** The reset time is determined by the slower of BSCK or TXC. If BSCK > TXC, T6 and T8 equal 10 TXCs. If BSCK < TXC, T6 and T8 equal 10 BSCKs (T3).

**Note 2:** These specifications are not tested.



## 7.0 AC and DC Specifications (Continued)

MEMORY WRITE, BMODE = 0, SYNCHRONOUS MODE (one wait-state shown)



TL/F/10492-59

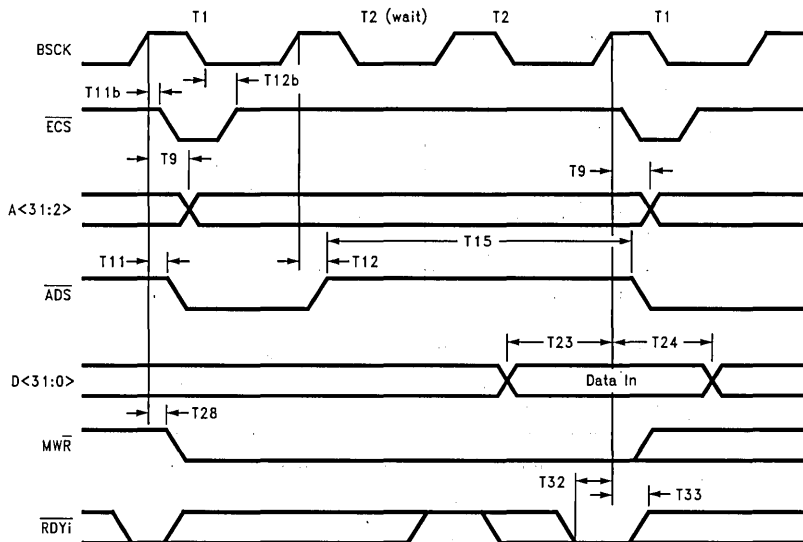
Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T9	BSCk to Address Valid/Hold Time	3	26	3	24	3	22	ns
T11	BSCk to $\overline{ADS}$ Low		26		24		22	ns
T11b	BSCk to $\overline{ECS}$ Low		19		17		15	ns
T12	BSCk to $\overline{ADS}$ High		24		22		20	ns
T12b	BSCk to $\overline{ECS}$ High		29		27		25	ns
T15	$\overline{ADS}$ High Width	45		35		25		ns
T32	$\overline{RDYi}$ Setup to BSCk	19		17		15		ns
T33	$\overline{RDYi}$ Hold from BSCk	5		3		3		ns
T36	BSCk to Memory Write Data Valid/Hold Time (Note 2)	3	50	3	48	3	46	ns
T37	BSCk to $\overline{MWR}$ (Write) Valid (Note 1)		24		22		20	ns

**Note 1:** For successive write operations,  $\overline{MWR}$  remains high.

**Note 2:** One idle clock cycle (T1) will be inserted between the last write cycle and the following read cycle in RDA and TDA operation. Note that the data bus will become TRI-STATE from the rising edge of the clock after the idle cycle (see T52 for BSCk to data TRI-STATE timing).

## 7.0 AC and DC Specifications (Continued)

MEMORY READ, BMODE = 0, SYNCHRONOUS MODE (one wait-state shown)



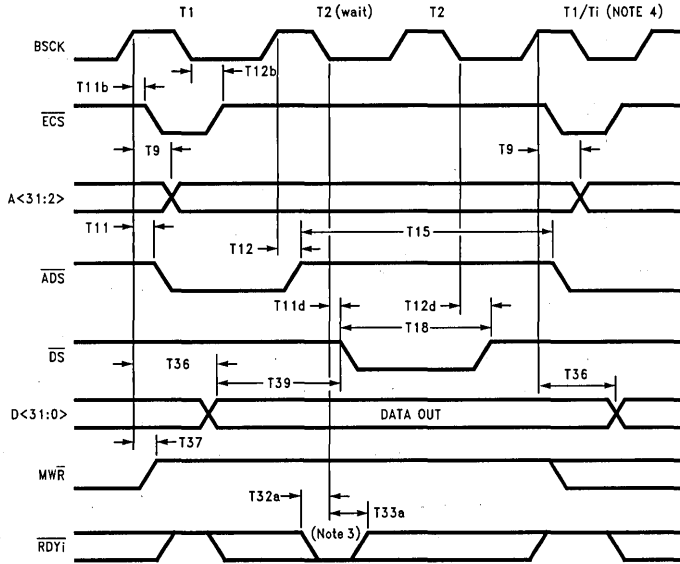
TL/F/10492-60

Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T9	BSCCK to Address Valid/Hold Time	3	26	3	24	3	22	ns
T11	BSCCK to $\overline{ADS}$ Low		26		24		22	ns
T11b	BSCCK to $\overline{ECS}$ Low		19		17		15	ns
T12	BSCCK to $\overline{ADS}$ High		24		22		20	ns
T12b	BSCCK to $\overline{ECS}$ High		29		27		25	ns
T15	$\overline{ADS}$ High Width	45		35		25		ns
T23	Read Data Setup Time to BSCCK	5		4		3		ns
T24	Read Data Hold Time to BSCCK	5		5		5		ns
T28	BSCCK to $\overline{MWR}$ (Read) Valid (Note 1)		26		24		22	ns
T32	$\overline{RDYi}$ Setup Time to BSCCK	19		17		15		ns
T33	$\overline{RDYi}$ Hold Time to BSCCK	5		3		3		ns

**Note 1:** For successive read operations,  $\overline{MWR}$  remains low.

## 7.0 AC and DC Specifications (Continued)

### MEMORY WRITE, BMODE = 0, ASYNCHRONOUS MODE



TL/F/10492-61

Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T9	BSCCK to Address Valid/Hold Time	3	26	3	24	3	22	ns
T11	BSCCK to $\overline{ADS}$ Low		26		24		22	ns
T11b	BSCCK to $\overline{ECS}$ Low		19		17		15	ns
T11d	BSCCK to $\overline{DS}$ Low		17		15		13	ns
T12	BSCCK to $\overline{ADS}$ High		24		22		20	ns
T12b	BSCCK to $\overline{ECS}$ High		29		27		25	ns
T12d	BSCCK to $\overline{DS}$ High		17		15		13	ns
T15	$\overline{ADS}$ High Width	45		35		25		ns
T18	Write Data Strobe Low Width (Note 2)	40		30		20		ns
T32a	$\overline{RDYi}$ Asynch. Setup to BSCCK (Note 3)	5		4		3		ns
T33a	$\overline{RDYi}$ Asynch. Hold from BSCCK	5		5		5		ns
T36	BSCCK to Memory Write Data Valid/Hold Time (Note 4)	3	50	3	48	3	46	ns
T37	BSCCK to $\overline{MWR}$ (Write) Valid (Note 1)		24		22		20	ns
T39	Write Data Valid to $\overline{DS}$ Low	34		21		7		ns

**Note 1:** For successive write operations,  $\overline{MWR}$  remains high.

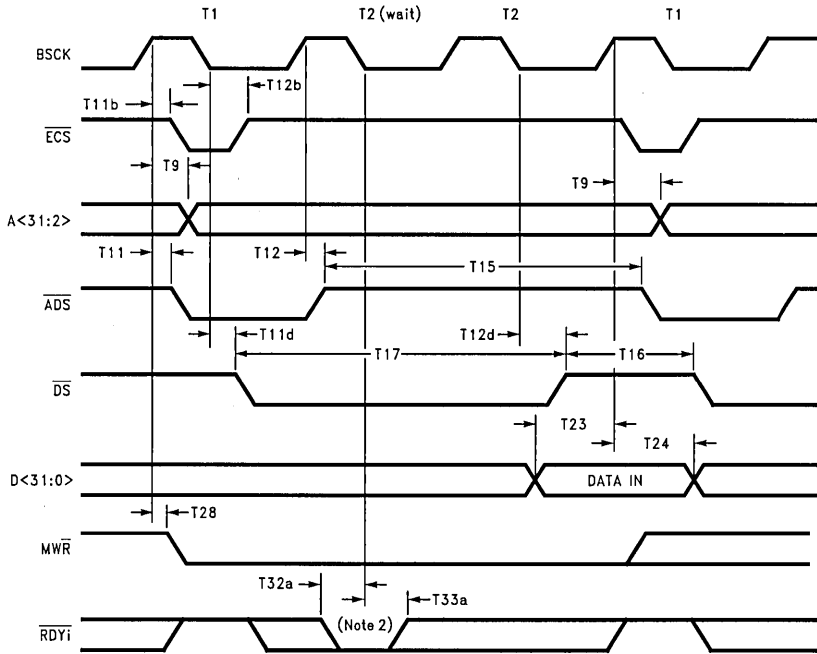
**Note 2:**  $\overline{DS}$  will only be asserted if the bus cycle has at least one wait state inserted.

**Note 3:** This setup time assures that the SONIC terminates the memory cycle on the next bus clock (BSCCK).  $\overline{RDYi}$  does not need to be synchronized to the bus clock, though, since it is an asynchronous input in this case.  $\overline{RDYi}$  is sampled during the falling edge of BSCCK. If the SONIC samples  $\overline{RDYi}$  low during the T1 cycle, the SONIC will finish the current access in a total of two bus clocks instead of three, which would be the case if  $\overline{RDYi}$  had been sampled low during T2(wait). (This is assuming that programmable wait states are set to 0).

**Note 4:** One idle clock cycle (T1) will be inserted between the last write cycle and the following read cycle in RDA and TDA operation. Note that the data bus will become TRI-STATE from the rising edge of the clock after the idle cycle (see T52 for BSCCK to data TRI-STATE timing).

## 7.0 AC and DC Specifications (Continued)

MEMORY READ, BMODE = 0, ASYNCHRONOUS MODE



TL/F/10492-62

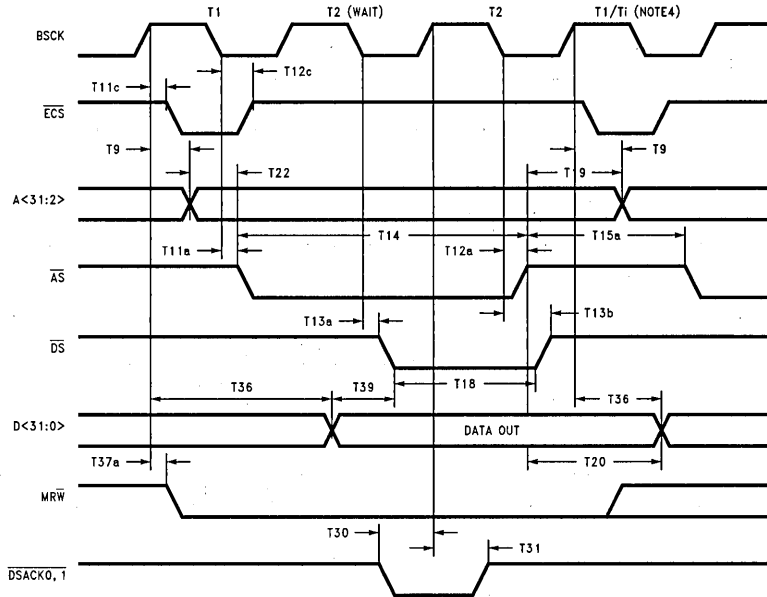
Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T9	BSCK to Address Valid/Hold Time	3	26	3	24	3	22	ns
T11	BSCK to $\overline{ADS}$ Low		26		24		22	ns
T11b	BSCK to $\overline{ECS}$ Low		19		17		15	ns
T11d	BSCK to $\overline{DS}$ Low		17		15		13	ns
T12	BSCK to $\overline{ADS}$ High		24		22		20	ns
T12b	BSCK to $\overline{ECS}$ High		29		27		25	ns
T12d	BSCK to $\overline{DS}$ High		17		15		13	ns
T15	$\overline{ADS}$ High Width	45		35		25		ns
T16	Read Data Strobe High Width	45		35		25		ns
T17	Read Data Strobe Low Width	40		30		20		ns
T23	Read Data Setup Time to BSCK	5		4		3		ns
T24	Read Data Hold Time from BSCK	5		5		5		ns
T28	BSCK to $\overline{MWR}$ (Read) Valid (Note 1)		26		24		22	ns
T32a	$\overline{RDYi}$ Asynch. Setup Time to BSCK (Note 2)	5		4		3		ns
T33a	$\overline{RDYi}$ Asynch. Hold Time to BSCK	5		5		5		ns

**Note 1:** For successive read operations,  $\overline{MWR}$  remains low.

**Note 2:** This setup time assures that the SONIC terminates the memory cycle on the next bus clock (BSCK).  $\overline{RDYi}$  does not need to be synchronized to the bus clock, though, since it is an asynchronous input in this case.  $\overline{RDYi}$  is sampled during the falling edge of BSCK. If the SONIC samples  $\overline{RDYi}$  low during the T1 cycle, the SONIC will finish the current access in a total of two bus clocks instead of three, which would be the case if  $\overline{RDYi}$  had been sampled low during T2(wait). (This is assuming that programmable wait states are set to 0).

## 7.0 AC and DC Specifications (Continued)

MEMORY WRITE, BMODE = 1, SYNCHRONOUS MODE (one wait-state shown)



TL/F/10492-63

Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T9	BSCCK to Address Valid/Hold Time	3	26	3	24	3	22	ns
T11a	BSCCK to $\overline{AS}$ Low		17		15		13	ns
T11c	BSCCK to $\overline{ECS}$ Low		19		17		15	ns
T12a	BSCCK to $\overline{AS}$ High		17		15		13	ns
T12c	BSCCK to $\overline{ECS}$ High		19		17		15	ns
T13a	BSCCK to $\overline{DS}$ Low (Note 1)		16		14		12	ns
T13b	BSCCK to $\overline{DS}$ High (Note 1)		16		14		12	ns
T14	$\overline{AS}$ Low Width	44		34		24		ns
T15a	$\overline{AS}$ High Width	45		35		25		ns
T18	Write Data Strobe Width (Note 1)	40		30		20		ns
T19	Address Hold Time from $\overline{AS}$	18		14		10		ns
T20	Data Hold Time from $\overline{AS}$	20		16		12		ns
T22	Address Valid to $\overline{AS}$ (Note 3)	9		6		2		ns
T30	$\overline{DSACK0,1}$ Setup to BSCCK (Note 3)	5		4		3		ns
T31	$\overline{DSACK0,1}$ Hold from BSCCK	9		8		7		ns
T36	BSCCK to Memory Write Data Valid/Hold Time (Note 4)	3	50	3	48	3	46	ns
T37a	BSCCK to MRW (Write) Valid (Note 2)		26		24		22	ns
T39	Write Data Valid to Data Strobe Low	34		21		7		ns

**Note 1:**  $\overline{DS}$  will only be asserted if the bus cycle has at least one wait state inserted.

**Note 2:** For successive write operations, MRW remains low.

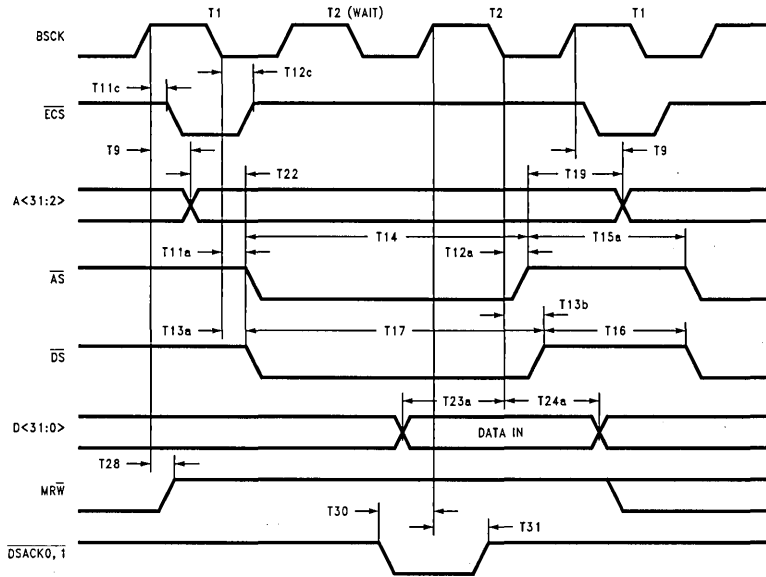
**Note 3:**  $\overline{DSACK0,1}$  must be synchronized to the bus clock (BSCCK) during synchronous mode.

**Note 4:** One idle clock cycle (T1) will be inserted between the last write cycle and the following read cycle in RDA and TDA operation. Note that the data bus will become TRI-STATE from the rising edge of the clock after the idle cycle (see T52 for BSCCK to data TRI-STATE timing).



## 7.0 AC and DC Specifications (Continued)

MEMORY READ, BMODE = 1, SYNCHRONOUS MODE (one wait-state shown)



TL/F/10492-64

Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T9	BSCCK to Address Valid	3	26	3	24	3	22	ns
T11a	BSCCK to $\overline{AS}$ Low		17		15		13	ns
T11c	BSCCK to $\overline{ECs}$ Low		19		17		15	ns
T12a	BSCCK to $\overline{AS}$ High		17		15		13	ns
T12c	BSCCK to $\overline{ECs}$ High		19		17		15	ns
T13a	BSCCK to $\overline{DS}$ Low (Note 3)		16		14		12	ns
T13b	BSCCK to $\overline{DS}$ High (Note 3)		16		14		12	ns
T14	$\overline{AS}$ Low Width	44		34		24		ns
T15a	$\overline{AS}$ High Width	45		35		25		ns
T16	Read Data Strobe High Width	45		35		25		ns
T17	Read Data Strobe Low Width	40		30		20		ns
T19	Address Hold Time from $\overline{AS}$	18		14		10		ns
T22	Address Valid to $\overline{AS}$	9		6		2		ns
T23a	Read Data Setup Time to BSCCK	5		4		3		ns
T24a	Read Data Hold Time from BSCCK	5		5		5		ns
T28	BSCCK to $\overline{MRW}$ (Read) Valid (Note 1)		26		24		22	ns
T30	$\overline{DSACK0,1}$ Setup to BSCCK (Note 2)	5		4		3		ns
T31	$\overline{DSACK0,1}$ Hold from BSCCK	9		8		7		ns

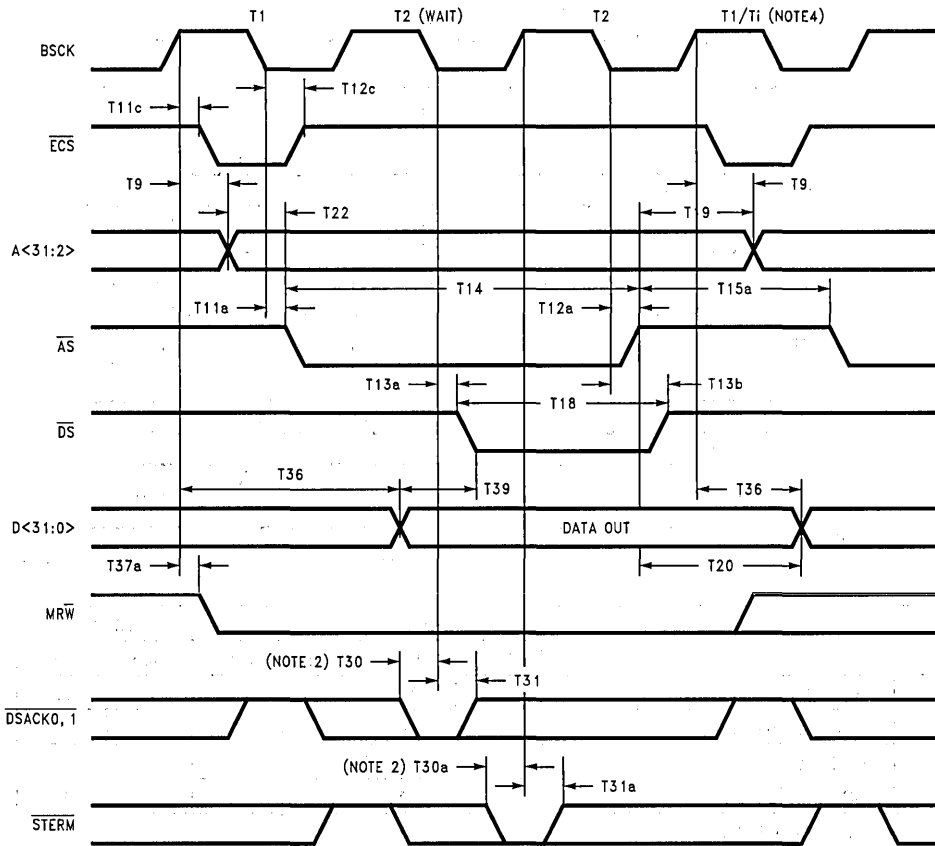
**Note 1:** For successive write operations,  $\overline{MRW}$  remains low.

**Note 2:**  $\overline{DSACK0,1}$  must be synchronized to the bus clock (BSCCK) during synchronized mode.

**Note 3:**  $\overline{DS}$  will only be asserted if the bus cycle has at last one wait state inserted.

### 7.0 AC and DC Specifications (Continued)

MEMORY WRITE, BMODE = 1, ASYNCHRONOUS MODE



TL/F/10492-65

## 7.0 AC and DC Specifications (Continued)

Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T9	BSCK to Address Valid	3	26	3	24	3	22	ns
T11a	BSCK to $\overline{AS}$ Low		17		15		13	ns
T11c	BSCK to $\overline{ECS}$ Low		19		17		15	ns
T12a	BSCK to $\overline{AS}$ High		17		15		13	ns
T12c	BSCK to $\overline{ECS}$ High		19		17		15	ns
T13a	BSCK to $\overline{DS}$ Low		16		14		12	ns
T13b	BSCK to $\overline{DS}$ High		16		14		12	ns
T14	$\overline{AS}$ Low Width	44		34		24		ns
T15a	$\overline{AS}$ High Width	45		35		25		ns
T18	Write Data Strobe Low Width (Note 3)	40		30		20		ns
T19	Address Hold Time from $\overline{AS}$	18		14		10		ns
T20	Data Hold Time from $\overline{AS}$	20		16		12		ns
T22	Address Valid to $\overline{AS}$	9		6		2		ns
T30	$\overline{DSACK0,1}$ Setup to BSCK (Note 2)	5		4		3		ns
T30a	$\overline{STERM}$ Setup to BSCK (Note 2)	5		4		3		ns
T31	$\overline{DSACK0,1}$ Hold from BSCK	9		8		7		ns
T31a	$\overline{STERM}$ Hold from BSCK	8		7		6		ns
T36	BSCK to Memory Write Data Valid (Note 4)	3	50	3	48	3	46	ns
T37a	BSCK to $\overline{MRW}$ (Write) Valid (Note 1)		26		24		22	ns
T39	Write Data Valid to Data Strobe Low	34		21		7		ns

**Note 1:** For successive write operations,  $\overline{MRW}$  remains low.

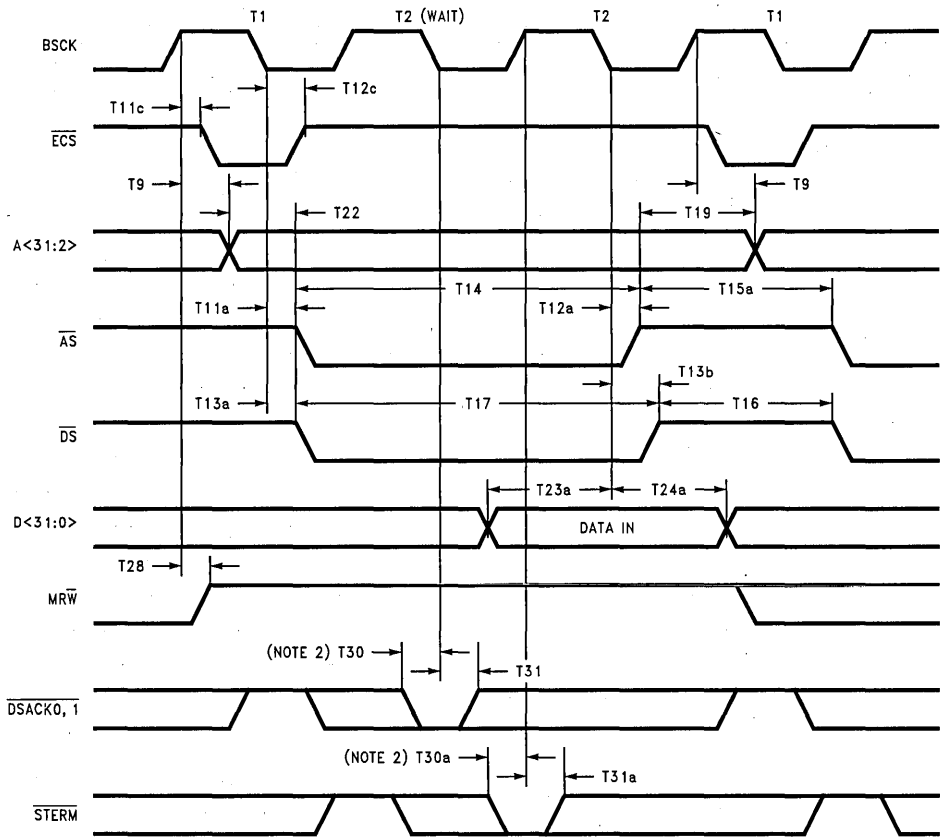
**Note 2:** Meeting the setup time for  $\overline{DSACK0,1}$  or  $\overline{STERM}$  guarantees that the SONIC will terminate the memory cycle  $1\frac{1}{2}$  bus clocks after  $\overline{DSACK0,1}$  were sampled, or 1 cycle after  $\overline{STERM}$  was sampled. T2 states will be repeated until  $\overline{DSACK0,1}$  or  $\overline{STERM}$  are sampled properly in a low state. If the SONIC samples  $\overline{DSACK0,1}$  or  $\overline{STERM}$  low during the T1 or first T2 state respectively, the SONIC will finish the current access in a total of two bus clocks instead of three (assuming that programmable wait states are set to 0),  $\overline{DSACK0,1}$  are asynchronously sampled and  $\overline{STERM}$  is synchronously sampled.

**Note 3:**  $\overline{DS}$  will only be asserted if the bus cycle has at least one wait state inserted.

**Note 4:** One idle clock cycle (T1) will be inserted between the last write cycle and the following read cycle in RDA and TDA operation. Note that the data bus will become TRI-STATE from the rising edge of the clock after the idle cycle (see T52 for BSCK to data TRI-STATE timing).

### 7.0 AC and DC Specifications (Continued)

#### MEMORY READ, BMODE = 1, ASYNCHRONOUS MODE



TL/F/10492-66

## 7.0 AC and DC Specifications (Continued)

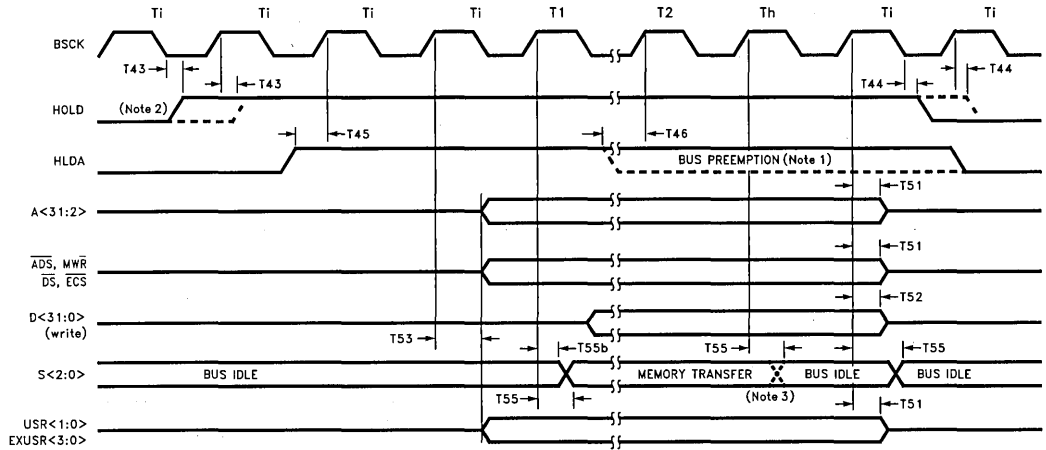
Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T9	BSCK to Address Valid	3	26	3	24	3	22	ns
T11a	BSCK to $\overline{AS}$ Low		17		15		13	ns
T11c	BSCK to $\overline{ECS}$ Low		19		17		15	ns
T12a	BSCK to $\overline{AS}$ High		17		15		13	ns
T12c	BSCK to $\overline{ECS}$ High		19		17		15	ns
T13a	BSCK to $\overline{DS}$ Low		16		14		12	ns
T13b	BSCK to $\overline{DS}$ High		16		14		12	ns
T14	$\overline{AS}$ Low Width	44		34		24		ns
T15a	$\overline{AS}$ High Width	45		35		25		ns
T16	Read Data Strobe High Width	45		35		25		ns
T17	Read Data Strobe Low Width	40		30		20		ns
T19	Address Hold Time from $\overline{AS}$	18		14		10		ns
T22	Address Valid to $\overline{AS}$	9		6		2		ns
T23a	Read Data Setup Time to BSCK	5		4		3		ns
T24a	Read Data Hold Time from BSCK	5		5		5		ns
T28	BSCK to $\overline{MRW}$ (Read) Valid (Note 1)		26		24		22	ns
T30	$\overline{DSACK0,1}$ Setup to BSCK (Note 2)	5		4		3		ns
T30a	$\overline{STERM}$ Setup to BSCK (Note 2)	5		4		3		ns
T31	$\overline{DSACK0,1}$ Hold from BSCK	9		8		7		ns
T31a	$\overline{STERM}$ Hold from BSCK	8		7		6		ns

**Note 1:** For successive read operations,  $\overline{MRW}$  remains high.

**Note 2:** Meeting the setup time for  $\overline{DSACK0,1}$  or  $\overline{STERM}$  guarantees that the SONIC will terminate the memory cycle  $1\frac{1}{2}$  bus clocks after  $\overline{DSACK0,1}$  were sampled, or 1 cycle after  $\overline{STERM}$  was sampled. T2 states will be repeated until  $\overline{DSACK0,1}$  or  $\overline{STERM}$  are sampled properly in a low state. If the SONIC samples  $\overline{DSACK0,1}$  or  $\overline{STERM}$  low during the T1 or first T2 state respectively, the SONIC will finish the current access in a total of two bus clocks instead of three (assuming that programmable wait states are set to 0).  $\overline{DSACK0,1}$  are asynchronously sampled and  $\overline{STERM}$  is synchronously sampled.

## 7.0 AC and DC Specifications (Continued)

### BUS REQUEST TIMING, BMODE = 0



TL/F/10492-67

Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T43	BSCK to HOLD High (Note 2)		18		16		14	ns
T44	BSCK to HOLD Low (Note 2)		19		17		15	ns
T45	HLDA Synchronous Setup Time to BSCK (Note 5)	7		6		5		ns
T46	HLDA Synchronous Deassert Setup Time (Note 1)	7		6		5		ns
T51	BSCK to Address, $\overline{ADS}$ , $\overline{MWR}$ , $\overline{DS}$ , $\overline{ECS}$ , $USR<1:0>$ and $EXUSR<3:0>$ TRI-STATE (Note 4)		34		32		30	ns
T52	BSCK to Data TRI-STATE		34		32		30	ns
T53	BSCK to $USR<1:0>$ or $EXUSR<3:0>$ Valid		34		32		30	ns
T55	BSCK to Bus Status Valid		29		27		25	ns
T55b	$S<2:0>$ Hold from BSCK	3		3		3		ns

**Note 1:** A block transfer by the SONIC can be pre-empted from the bus by deasserting HLDA provided HLDA is deasserted T46 before the rising edge of the last T2 in the current access.

**Note 2:** The assertion edge for HOLD is dependent upon the PH bit in the DCR2. The default situation is shown with a solid line in the timing diagram. T43 and T44 apply for both modes. Also, if HLDA is asserted when the SONIC wants to acquire the bus, HOLD will not be asserted until HLDA has been deasserted first.

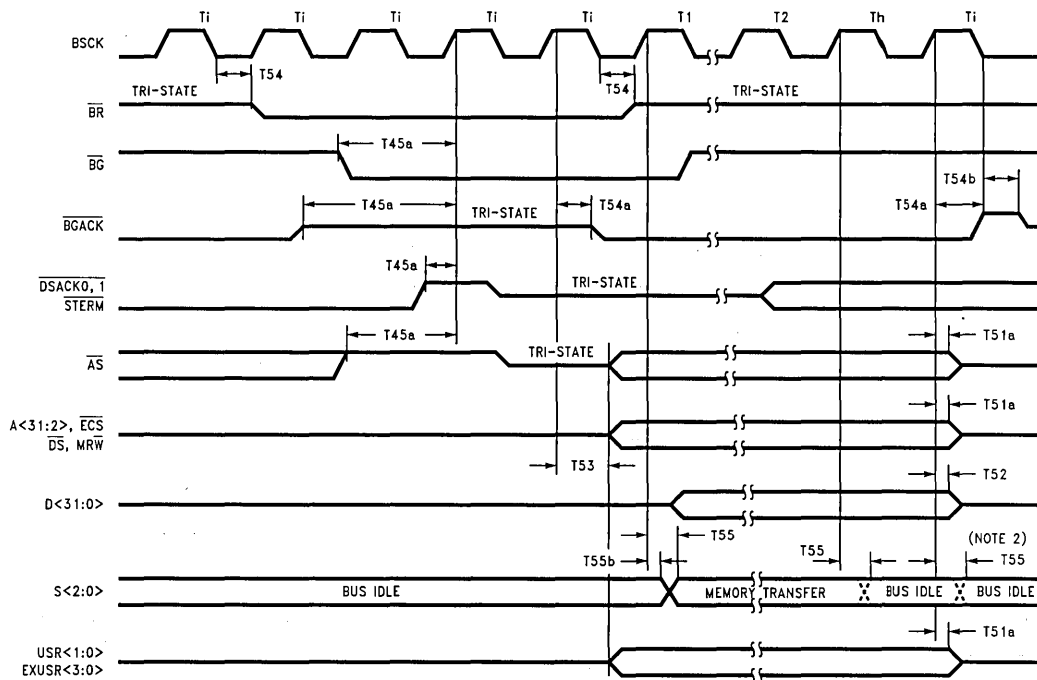
**Note 3:**  $S<2:0>$  will indicate IDLE at the end of T2 if the last operation is a read operation, or at the end of Th if the last operation is a write operation.

**Note 4:** This timing value includes an RC delay inherent in the test measurement. These signals typically TRI-STATE 7 ns earlier, enabling other devices to drive these lines without contention.

**Note 5:** The HLDA signal is sampled by the SONIC on each rising edge of BSCK. The maximum setup time is  $((BSCK\_period - T45\_min\_spec) - 5ns)$ . The HLDA max setup time is for information only, and is not tested.

## 7.0 AC and DC Specifications (Continued)

### BUS REQUEST TIMING, BMODE = 1



TL/F/10492-68

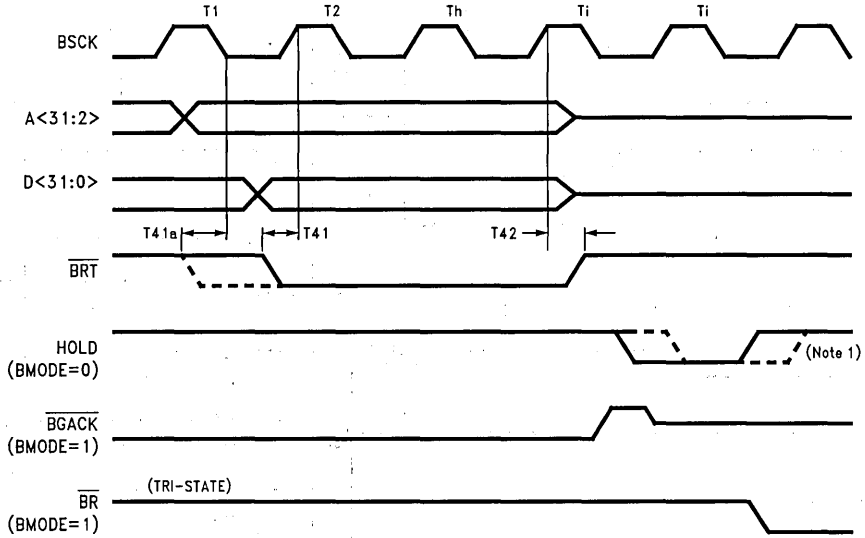
Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T45a	$\overline{BG}$ , $\overline{AS}$ , $\overline{BGACK}$ , $\overline{DSACK0,1}$ , and $\overline{STERM}$ Asynchronous Setup Time to BSCK (Note 1)	7		6		5		ns
T51a	BSCK to Address, $\overline{AS}$ , $\overline{MRW}$ , $\overline{DS}$ , $\overline{ECS}$ , $\overline{USR}<1:0>$ and $\overline{EXUSR}<3:0>$ TRI-STATE		34		32		30	ns
T52	BSCK to Data TRI-STATE		34		32		30	ns
T53	BSCK to Address, $\overline{AS}$ , $\overline{MRW}$ , $\overline{DS}$ , $\overline{ECS}$ , $\overline{USR}<1:0>$ , and $\overline{EXUSR}<3:0>$ Active (Note 1)		34		32		30	ns
T54	BSCK Low to $\overline{BR}$ Low/TRI-STATE		23		21		19	ns
T54a	BSCK High to $\overline{BGACK}$ Low/High		24		22		20	ns
T54b	BSCK High to $\overline{BGACK}$ TRI-STATE		19		17		15	ns
T55	BSCK to Bus Status Valid		29		27		25	ns
T55b	$S<2:0>$ Hold from BSCK	3		3		3		ns

**Note 1:**  $\overline{BGACK}$  is asserted one bus clock after all the signals ( $\overline{AS}$ ,  $\overline{DSACK0,1}$ ,  $\overline{BGACK}$ ,  $\overline{STERM}$  (Extended bus mode), and  $\overline{BG}$ ) meet the T45a setup time (see Section 5.4.1 for more information). The address bus,  $\overline{AS}$ ,  $\overline{DS}$ ,  $\overline{ECS}$ ,  $\overline{MRW}$ ,  $\overline{USR}<1:0>$ , and  $\overline{EXUSR}<3:0>$  will also be driven active on the same clock.

**Note 2:**  $S<2:0>$  will indicate IDLE at the end of T2 if the last operation is a read operation, or at the end of Th if the last operation is a write operation.

## 7.0 AC and DC Specifications (Continued)

### BUS RETRY



TL/F/10492-69

Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T41	Bus Retry Synchronous Setup Time to BSCCK (Note 3)	5		4		3		ns
T41a	Bus Retry Asynchronous Setup Time to BSCCK (Note 3)	6		5		4		ns
T42	Bus Retry Hold Time from BSCCK (Note 2)	7		6		5		ns

**Note 1:** Depending upon the mode, the SONIC will assert and deassert HOLD from the rising or falling edge of BSCCK.

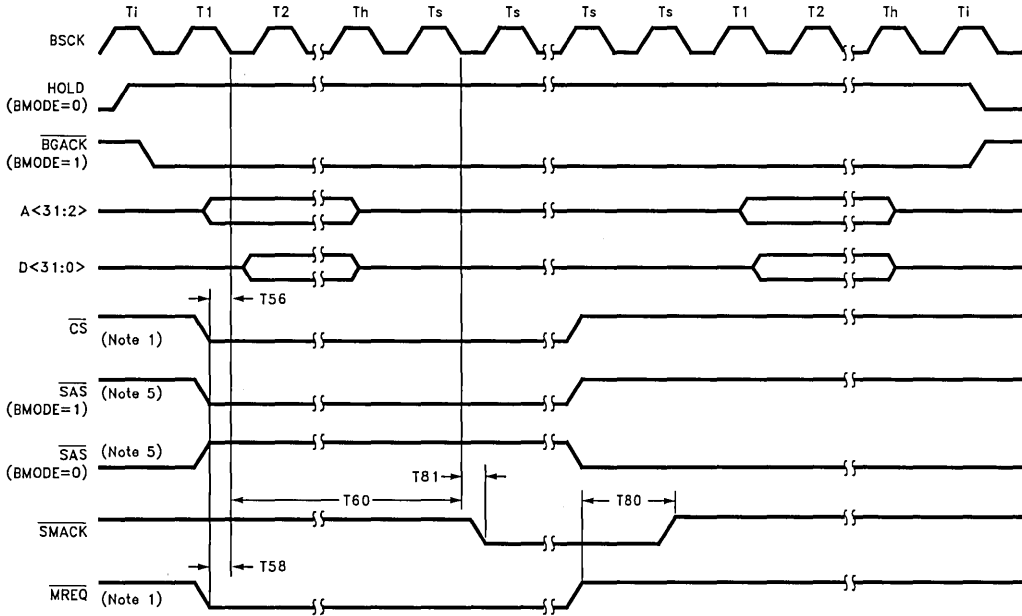
**Note 2:** Unless Latched Bus Retry mode is set (LBR in the Data Configuration Register, Section 4.3.2), BRT must remain asserted until after the Th state. If Latched Bus Retry mode is used, BRT does not need to satisfy T42.

**Note 3:** T41 is for synchronous bus retry and T41a is for asynchronous bus retry (see Section 4.3.2, bit 15, Extended Bus Mode). Since T41a is an asynchronous setup time, it is not necessary to meet it, but doing so will guarantee that the bus exception occurs in the current memory transfer, not the next.



## 7.0 AC and DC Specifications (Continued)

### MEMORY ARBITRATION/S�AVE ACCESS



TL/F/10492-70

Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T56	$\overline{CS}$ Low Asynch. Setup to BSCk (Note 2)	8		7		6		ns
T58	$\overline{MREQ}$ Low Asynch. Setup to BSCk (Note 2)	8		7		6		ns
T60	$\overline{MREQ}$ or $\overline{CS}$ Valid to $\overline{SMACK}$ Low (Notes 3, 4)	1	5	1	5	1	5	bcyc
T80	$\overline{MREQ}$ to $\overline{SMACK}$ High		18		16		14	ns
T81	BSCk to $\overline{SMACK}$ Low		22		20		18	ns

**Note 1:** Both  $\overline{CS}$  and  $\overline{MREQ}$  must not be asserted concurrently. If these signals are successively asserted, there must be at least two bus clocks between the deasserting and asserting edges of these signals.

**Note 2:** It is not necessary to meet the setup times for  $\overline{MREQ}$  or  $\overline{CS}$  since these signals are asynchronously sampled. Meeting the setup time for these signals, however, makes it possible to use T60 to determine when  $\overline{SMACK}$  will be asserted.

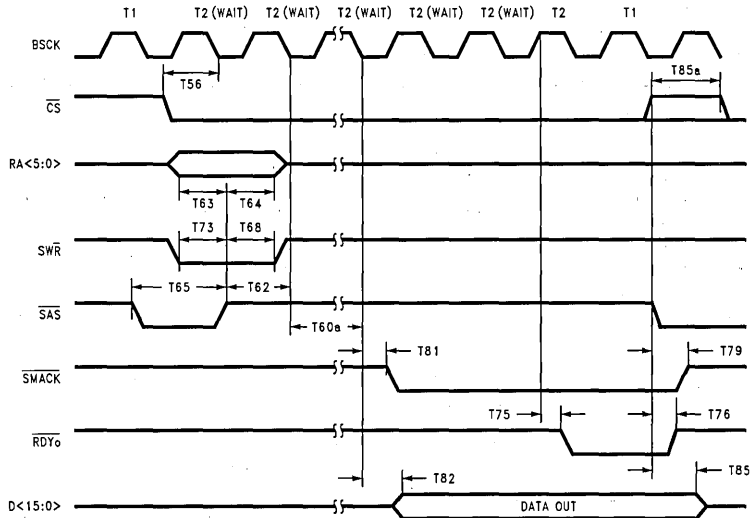
**Note 3:** T60 could range from 1 bus clock minimum to 5 bus clock maximum depending on what state machine the SONIC is in when the  $\overline{CS}$  or  $\overline{MREQ}$  signal is asserted. This timing is not tested, but is guaranteed by design. This specification assumes that  $\overline{CS}$  or  $\overline{MREQ}$  is asserted before the falling edge that these signals are asynchronously clocked in on (see T56 and T58).  $\overline{SAS}$  must have been asserted for this timing to be correct. See  $\overline{SAS}$  and  $\overline{CS}$  timing in the Register Read, and Register Write timing specifications.

**Note 4:** bcyc = bus clock cycle time (T3).

**Note 5:** The way in which  $\overline{SMACK}$  is asserted due to  $\overline{CS}$  is not the same as the way in which  $\overline{SMACK}$  is asserted due to  $\overline{MREQ}$ .  $\overline{SMACK}$  goes low as a direct result of the assertion of  $\overline{MREQ}$ , whereas, for  $\overline{CS}$ ,  $\overline{SAS}$  must also be driven low (BMODE = 1) or high (BMODE = 0) before  $\overline{SMACK}$  will be asserted. This means that when  $\overline{SMACK}$  is asserted due to  $\overline{MREQ}$ ,  $\overline{SMACK}$  will remain asserted until  $\overline{MREQ}$  is deasserted. Multiple memory accesses can be made to the shared memory without  $\overline{SMACK}$  ever going high. When  $\overline{SMACK}$  is asserted due to  $\overline{CS}$ , however,  $\overline{SMACK}$  will only remain low as long as  $\overline{SAS}$  is also low (BMODE = 1) or high (BMODE = 0).  $\overline{SMACK}$  will not remain low throughout multiple register accesses to the SONIC because  $\overline{SAS}$  must toggle for each register access. This is an important difference to consider when designing shared memory designs.

## 7.0 AC and DC Specifications (Continued)

### REGISTER READ, BMODE = 0 (Note 1)



TL/F/10492-88

Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T56	$\overline{CS}$ Asynch. Setup to BSCCK (Notes 4, 6)	8		7		6		ns
T60a	$\overline{CS}$ and $\overline{SAS}$ to $\overline{SMACK}$ Low (Notes 3, 5, 6)	0	4	0	4	0	4	bcyk
T62	$\overline{SAS}$ Asynch. Setup to BSCCK (Notes 4, 6)	7		6		5		ns
T63	Register Address Setup Time to $\overline{SAS}$	7		6		5		ns
T64	Register Address Hold Time from $\overline{SAS}$	8		7		6		ns
T65	Minimum $\overline{SAS}$ Low Width (Notes 4, 6)	20		17		15		ns
T68	$\overline{SWR}$ (Read) Hold from $\overline{SAS}$	8		7		6		ns
T73	$\overline{SWR}$ (Read) Setup to $\overline{SAS}$	7		6		5		ns
T75	BSCCK to $\overline{RDY0}$ Low		20		18		16	ns
T76	$\overline{SAS}$ or $\overline{CS}$ to $\overline{RDY0}$ High (Note 2)		34		32		30	ns
T79	$\overline{SAS}$ or $\overline{CS}$ to $\overline{SMACK}$ High (Note 2)		18		16		14	ns
T81	BSCCK to $\overline{SMACK}$ Low		22		20		18	ns
T82	BSCCK to Register Data Valid		44		42		40	ns
T85	$\overline{SAS}$ or $\overline{CS}$ to Data TRI-STATE (Notes 2, 7)		34		32		30	ns
T85a	Min. $\overline{CS}$ Deassert Time (Note 3)	1		1		1		bcyk

**Note 1:** This figure shows a slave access to the SONIC. The BSCCK states (T1, T2, etc.) are the equivalent processor states during a slave access.

**Note 2:** If  $\overline{CS}$  is deasserted before the falling edge of  $\overline{SAS}$ , T76, T79 and T85 are referenced from the rising edge of  $\overline{CS}$ .

**Note 3:** bcyk = bus clock cycle time (T3).

**Note 4:** It is not necessary to meet the setup time for  $\overline{CS}$  (T56) and the setup time for  $\overline{SAS}$  (T62) since these signals are asynchronously sampled. Meeting these setup times for these signals, however, makes it possible to use T60a to determine exactly when  $\overline{SMACK}$  will be asserted. For multiple register accesses,  $\overline{CS}$  can be held low and  $\overline{SAS}$  can be used to delimit the slave cycle. In this case,  $\overline{SMACK}$  will be driven low by the SONIC after T60a when T62 is met. T85a must be met to ensure proper slave operation once  $\overline{CS}$  is deasserted.

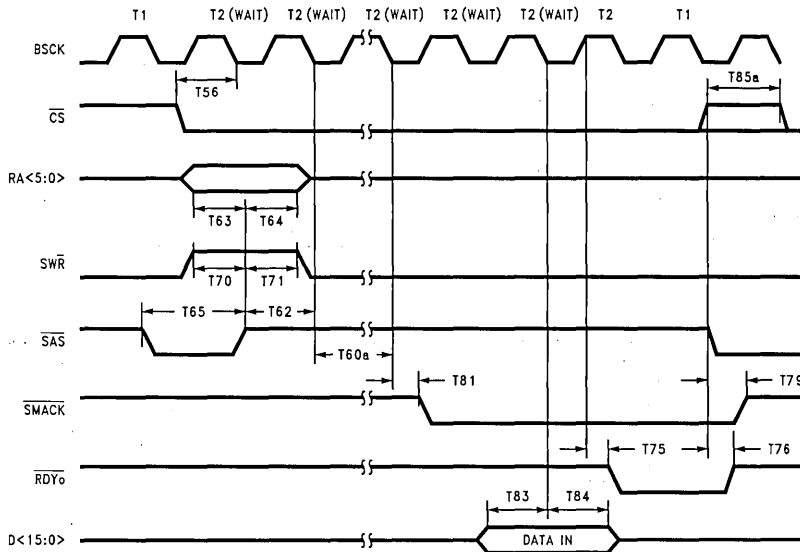
**Note 5:** The smaller value for T60a refers to when the SONIC is accessed during an Idle condition and the other value refers to when the SONIC is accessed during non-idle conditions. These values are not tested, but are guaranteed by design.

**Note 6:**  $\overline{SAS}$  may be asserted low anytime before or simultaneous to the falling edge of  $\overline{CS}$ . Register address and slave read/write signals are latched on the rising edge of the  $\overline{SAS}$ , and if T62 is met,  $\overline{SMACK}$  will be asserted by the SONIC after T60a. If T62 is not met, SONIC will sample  $\overline{SAS}$  again on the next falling edge of the clock, and  $\overline{SMACK}$  will not be asserted until  $\overline{SAS}$  is deasserted.

**Note 7:** This timing value includes an RC delay inherent in the test measurement. These signals typically TRI-STATE 7 ns earlier, enabling other devices to drive these lines without contention.

## 7.0 AC and DC Specifications (Continued)

REGISTER WRITE, BMODE = 0 (Note 1)



TLF/10492-89

Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T56	$\overline{CS}$ Asynch. Setup to BCLK (Notes 4, 6)	8		7		6		ns
T60a	$\overline{CS}$ and $\overline{SAS}$ to $\overline{SMACK}$ Low (Notes 3, 5, 6)	0	4	0	4	0	4	bcyc
T62	$\overline{SAS}$ Asynch. Setup to BCLK (Notes 4, 6)	7		6		5		ns
T63	Register Address Setup Time to $\overline{SAS}$	7		6		5		ns
T64	Register Address Hold Time from $\overline{SAS}$	8		7		6		ns
T65	Minimum $\overline{SAS}$ Low Width (Notes 4, 6)	20		17		15		ns
T70	$\overline{SWR}$ (Write) Setup to $\overline{SAS}$	7		6		5		ns
T71	$\overline{SWR}$ (Write) Hold from $\overline{SAS}$	8		7		6		ns
T75	BCLK to $\overline{RDY0}$ Low		20		18		16	ns
T76	$\overline{SAS}$ or $\overline{CS}$ to $\overline{RDY0}$ High (Note 2)		34		32		30	ns
T79	$\overline{SAS}$ or $\overline{CS}$ to $\overline{SMACK}$ High (Note 2)		18		16		14	ns
T81	BCLK to $\overline{SMACK}$ Low		22		20		18	ns
T83	Register Write Data Setup to BCLK	7		6		5		ns
T84	Register Write Data Hold from BCLK	14		12		10		ns
T85a	Min. $\overline{CS}$ Deassert Time (Note 3)	1		1		1		bcyc

**Note 1:** This figure shows a slave access to the SONIC. The BCLK states (T1, T2, etc.) are the equivalent processor states during a slave access.

**Note 2:** If  $\overline{CS}$  is deasserted before the falling edge of  $\overline{SAS}$ , T76, T79 and T85 are referenced from the rising edge of  $\overline{CS}$ .

**Note 3:** bcyc = bus clock cycle time (T3).

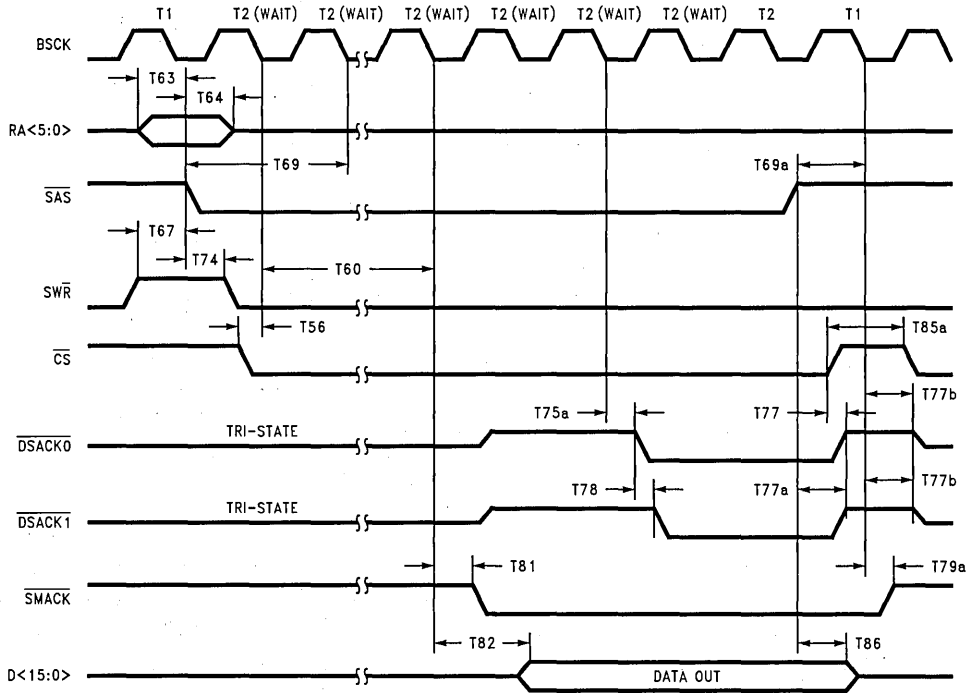
**Note 4:** It is not necessary to meet the setup time for  $\overline{CS}$  (T56) and the setup time for  $\overline{SAS}$  (T62) since these signals are asynchronously sampled. Meeting these setup times for these signals, however, makes it possible to use T60a to determine exactly when  $\overline{SMACK}$  will be asserted. For multiple register accesses,  $\overline{CS}$  can be held low and  $\overline{SAS}$  can be used to delimit the slave cycle. In this case,  $\overline{SMACK}$  will be driven low by the SONIC after T60a when T62 is met. T85a must be met to ensure proper slave operation once  $\overline{CS}$  is deasserted.

**Note 5:** The smaller value for T60a refers to when the SONIC is accessed during an Idle condition and the other value refers to when the SONIC is accessed during non-idle conditions. These values are not tested, but are guaranteed by design.

**Note 6:**  $\overline{SAS}$  may be asserted low anytime before or simultaneous to the falling edge of  $\overline{CS}$ . Register address and slave read/write signals are latched on the rising edge of the  $\overline{SAS}$ , and if T62 is met,  $\overline{SMACK}$  will be asserted by the SONIC after T60a. If T62 is not met, SONIC will sample  $\overline{SAS}$  again on the next falling edge of the clock, and  $\overline{SMACK}$  will not be asserted until  $\overline{SAS}$  is deasserted.

### 7.0 AC and DC Specifications (Continued)

REGISTER READ, BMODE = 1 (Note 1)



TL/F/10492-90

## 7.0 AC and DC Specifications (Continued)

Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T56	$\overline{CS}$ Asynch. Setup to BSCK (Notes 3, 4)	8		7		6		ns
T60	$\overline{CS}$ Valid to $\overline{SMACK}$ Low (Notes 2, 3, 4)	1	5	1	5	1	5	bcyc
T63	Register Address Setup to $\overline{SAS}$	6		5		4		ns
T64	Register Address Hold from $\overline{SAS}$	8		7		6		ns
T67	$SR\overline{W}$ (Read) Setup to $\overline{SAS}$	4		3		2		ns
T69	$\overline{SAS}$ Asynch. Setup to BSCK (Notes 3, 4)	7		6		5		ns
T69a	$\overline{SAS}$ Asynch. Setup to BSCK (Notes 3, 5)	5		4		3		ns
T74	$SR\overline{W}$ (Read) Hold from $\overline{SAS}$	8		7		6		ns
T75a	BSCK to $\overline{DSACK0,1}$ Low		14		12		10	ns
T77	$\overline{CS}$ to $\overline{DSACK0,1}$ High (Note 5)		20		18		16	ns
T77a	$\overline{SAS}$ to $\overline{DSACK0,1}$ High (Note 5)		24		22		20	ns
T77b	BSCK to $\overline{DSACK0,1}$ TRI-STATE (Note 5)		19		17		15	ns
T78	Skew between $\overline{DSACK0,1}$		3		3		2	ns
T79a	BSCK to $\overline{SMACK}$ High (Note 5)		19		17		15	ns
T81	BSCK to $\overline{SMACK}$ Low		22		20		18	ns
T82	BSCK to Register Data Valid		44		42		40	ns
T85a	Min. $\overline{CS}$ Deassert Time (Notes 2, 3)	1		1		1		bcyc
T86	$\overline{SAS}$ to Register Data TRI-STATE (Note 6)		42		40		38	ns

**Note 1:** This figure shows a slave access to the SONIC when the SONIC is idle, or rather not in master mode. If the SONIC is a bus master, there will be some differences as noted in the Memory Arbitration/Slave Access diagram. The BSCK states (T1, T2, etc.) are the equivalent processor states during a slave access.

**Note 2:** bcyc = bus clock cycle time (T3).

**Note 3:** It is not necessary to meet the setup time for  $\overline{CS}$  and  $\overline{SAS}$  (T56 and T69) since these signals are asynchronously sampled. Meeting the setup time for these signals, however, makes it possible to use T60 to determine when  $\overline{SMACK}$  will be asserted.  $\overline{SAS}$  may be asserted anytime before the next falling edge of the clock that the  $\overline{CS}$  is sampled on (as shown by specification T69). For multiple register accesses,  $\overline{CS}$  can be held low and  $\overline{SAS}$  can be used to delimit the slave cycle (T69a must be met in order to terminate and start another cycle). In this case,  $\overline{SMACK}$  will be asserted as soon as T69 timing is met.

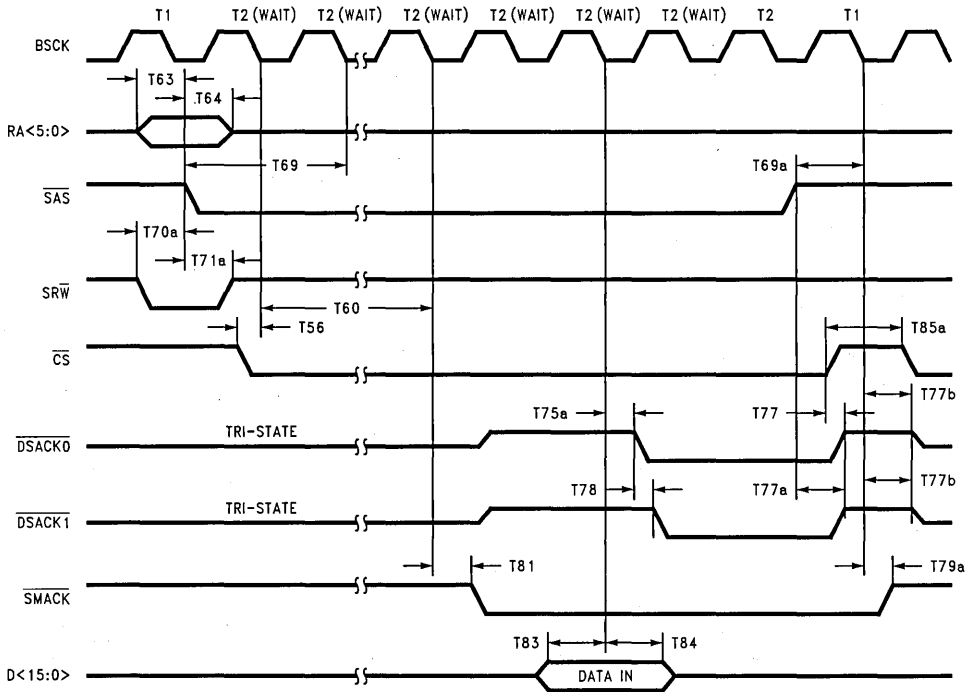
**Note 4:** T60 could range from 1 bus clock minimum to 5 bus clock maximum depending on what state machine the SONIC is in when the  $\overline{CS}$  signal is asserted. This timing is not tested, but is guaranteed by design. This specification assumes that both T56 is met for  $\overline{CS}$  and T69 is met for  $\overline{SAS}$ . T60 specification also assumes that there were no wait states in the current master mode access (if  $\overline{CS}$  is asserted when SONIC is in Master Mode). If there were wait states, then it would increase the T60 further.

**Note 5:** It is not necessary to meet the setup times for  $\overline{SAS}$  (T69a) since this signal is asynchronously sampled. Meeting the setup time for this signal, however, will ensure  $\overline{DSACK0,1}$  becomes TRI-STATE (T77b) and  $\overline{SMACK}$  goes high (T79) at the falling edge of T1. Both  $\overline{CS}$  and  $\overline{SAS}$  could cause  $\overline{DSACK0,1}$  to deassert but only  $\overline{SAS}$  could cause  $\overline{DSACK0,1}$  to become TRI-STATE.

**Note 6:** This timing value includes an RC delay inherent in the test measurement. These signals typically TRI-STATE 7 ns earlier, enabling other devices to drive these lines without contention.

### 7.0 AC and DC Specifications (Continued)

REGISTER WRITE, BMODE = 1 (Note 1)



TL/F/10492-74

## 7.0 AC and DC Specifications (Continued)

Number	Parameter	20 MHz		25 MHz		33 MHz		Units
		Min	Max	Min	Max	Min	Max	
T56	$\overline{CS}$ Asynch. Setup to BSCK (Notes 3, 4)	8		7		6		ns
T60	$\overline{CS}$ valid to $\overline{SMACK}$ Low (Notes 2, 3, 4)	1	5	1	5	1	5	bcyc
T63	Register Address Setup to $\overline{SAS}$	6		5		4		ns
T64	Register Address Hold from $\overline{SAS}$	8		7		6		ns
T69	$\overline{SAS}$ Asynch. Setup to BSCK (Notes 3, 4)	7		6		5		ns
T69a	$\overline{SAS}$ Asynch. Setup to BSCK (Notes 3, 5)	5		4		3		ns
T70a	$\overline{SRW}$ (Write) Setup to $\overline{SAS}$	4		3		2		ns
T71a	$\overline{SRW}$ (Write) Hold from $\overline{SAS}$	8		7		6		ns
T75b	BSCK to $\overline{DSACK0,1}$ Low		14		12		10	ns
T77	$\overline{CS}$ to $\overline{DSACK0,1}$ High (Note 5)		20		18		16	ns
T77a	$\overline{SAS}$ to $\overline{DSACK0,1}$ High (Note 5)		24		22		20	ns
T77b	BSCK to $\overline{DSACK0,1}$ TRI-STATE (Note 5)		19		17		15	ns
T78	Skew between $\overline{DSACK0,1}$		3		3		2	ns
T79a	BSCK to $\overline{SMACK}$ High (Note 5)		19		17		15	ns
T81	BSCK to $\overline{SMACK}$ Low		22		20		18	ns
T83	Register Write Data Setup to BSCK	7		6		5		ns
T84	Register Write Data Hold from BSCK	14		12		10		ns
T85a	Min. $\overline{CS}$ Deassert Time (Notes 2, 3)	1		1		1		bcyc

**Note 1:** This figure shows a slave access to the SONIC when the SONIC is idle, or rather not in master mode. If the SONIC is a bus master, there will be some differences as noted in the Memory Arbitration/Slave Access diagram. The BSCK states (T1, T2, etc.) are the equivalent processor states during a slave access.

**Note 2:** bcyc = bus clock cycle time (T3).

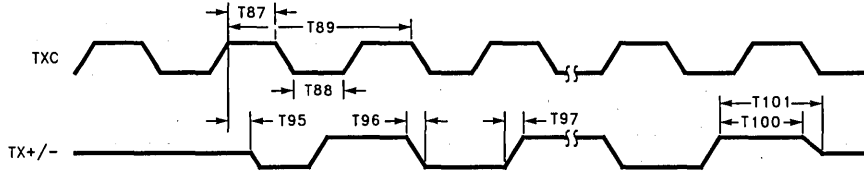
**Note 3:** It is not necessary to meet the setup time for  $\overline{CS}$  and  $\overline{SAS}$  (T56 and T69) since these signals are asynchronously sampled. Meeting the setup time for these signals, however, makes it possible to use T60 to determine when  $\overline{SMACK}$  will be asserted.  $\overline{SAS}$  may be asserted anytime before the next falling edge of the clock that the  $\overline{CS}$  is sampled on (as shown by specification T69). For multiple register accesses,  $\overline{CS}$  can be held low and  $\overline{SAS}$  can be used to delimit the slave cycle (T69a must be met in order to terminate and start another cycle). In this case,  $\overline{SMACK}$  will be asserted as soon as T69 timing is met.

**Note 4:** T60 could range from 1 bus clock minimum to 5 bus clock maximum depending on what state machine the SONIC is in when the  $\overline{CS}$  signal is asserted. This timing is not tested, but is guaranteed by design. This specification assumes that both T56 is met for  $\overline{CS}$  and T69 is met for  $\overline{SAS}$ . T60 specification also assumes that there were no wait states in the current master mode access (if  $\overline{CS}$  is asserted when SONIC is in Master Mode). If there were wait states, then it would increase the T60 further.

**Note 5:** It is not necessary to meet the setup time for  $\overline{SAS}$  (T69a) since this signal is asynchronously sampled. Meeting the setup time for this signal, however, will ensure  $\overline{DSACK0,1}$  becomes TRI-STATE (T77b) and  $\overline{SMACK}$  goes high (T79) at the falling edge of T1. Both  $\overline{CS}$  and  $\overline{SAS}$  could cause  $\overline{DSACK0,1}$  to deassert but only  $\overline{SAS}$  could cause  $\overline{DSACK0,1}$  to become TRI-STATE.

## 7.0 AC and DC Specifications (Continued)

### ENDEC TRANSMIT TIMING



TL/F/10492-75

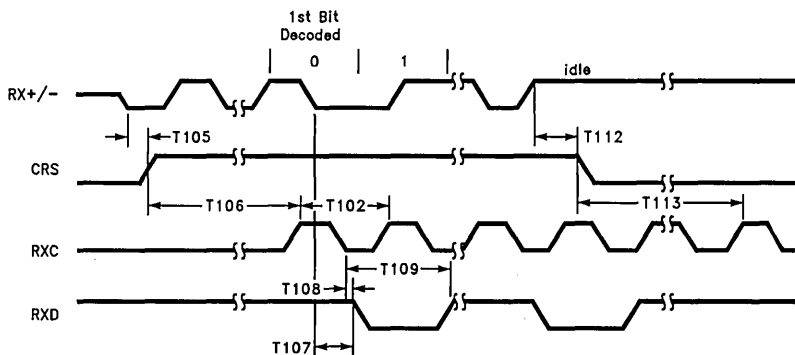
Number	Parameter	Min	Max	Units
T87	Transmit Clock High Time (Note 1)	40		ns
T88	Transmit Clock Low Time (Note 1)	40		ns
T89	Transmit Clock Cycle Time (Note 1)	99.99	100.01	ns
T95	Transmit Output Delay (Note 1)		55	ns
T96	Transmit Output Fall Time (80% to 20%, Note 1)		7	ns
T97	Transmit Output Rise Time (20% to 80%, Note 1)		7	ns
T98	Transmit Output Jitter (Not Shown)	0.5 Typical		ns
T100	Transmit Output High before Idle (Half Step)	200		ns
T101	Transmit Output Idle Time (Half Step)		8000	ns

Note 1: This specification is provided for information only and is not tested.



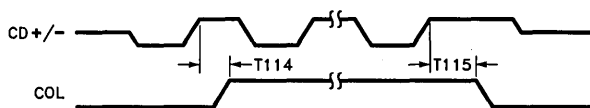
## 7.0 AC and DC Specifications (Continued)

### ENDEC RECEIVE TIMING (INTERNAL ENDEC MODE)



TL/F/10492-76

### ENDEC COLLISION TIMING



TL/F/10492-77

Number	Parameter	Min	Max	Units
T102	Receive Clock Duty Cycle Time (Note 1)	40	60	ns
T105	Carrier Sense On Time		70	ns
T106	Data Acquisition Time		700	ns
T107	Receive Data Output Delay		150	ns
T108	Receive Data Valid from RXC		15	ns
T109	Receive Data Stable Valid Time	85		ns
T112	Carrier Sense Off Delay (Note 2)		250	ns
T113	Minimum Number of RXCs after CRS Low (Note 3)	5		rcyc
T114	Collision Turn On Time		55	ns
T115	Collision Turn Off Time		250	ns

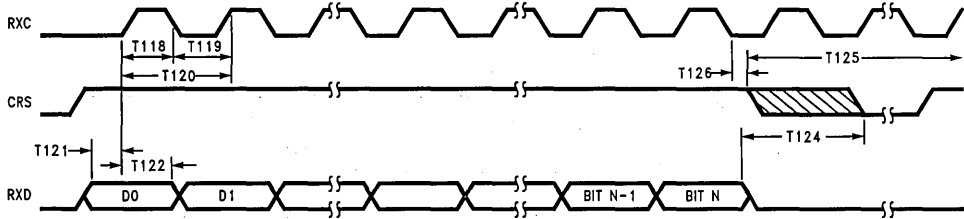
**Note 1:** This parameter is measured at the 50% point of each clock edge.

**Note 2:** When CRSi goes low, it remains low for a minimum of 2 receive clocks (RXC).

**Note 3:** rcyc = receive clocks.

## 7.0 AC and DC Specifications (Continued)

### ENDEC-MAC SERIAL TIMING FOR RECEPTION (EXTERNAL ENDEC MODE)



TL/F/10492-78

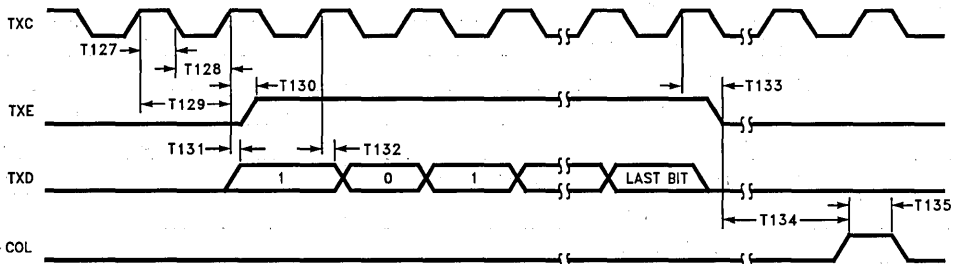
Number	Parameter	Min	Max	Units
T118	Receive Clock High Time	40		ns
T119	Receive Clock Low Time	40		ns
T120	Receive Clock Cycle Time	90	110	ns
T121	RXD Setup to RXC	20		ns
T122	RXD Hold from RXC	15		ns
T124	Maximum Allowed Dribble Bits		6	Bits
T125	Receive Recovery Time (Note 2)			
T126	RXC to Carrier Sense Low (Notes 1, 3)		1	rcyc

**Note 1:**  $t_{cyc}$  = transmit clocks,  $r_{cyc}$  = receive clocks,  $b_{cyc}$  = T3.

**Note 2:** This parameter refers to longest time (not including wait-states) the SONIC requires to perform its end of receive processing and be ready for the next start of frame delimiter. This time is  $4 + 36 t_{cyc} b_{cyc}$ . This is guaranteed by design and is not tested.

**Note 3:** To ensure proper receive operation, a minimum of 5 RXCs after CRS low are required.

### ENDEC-MAC SERIAL TIMING FOR TRANSMIT (NO COLLISION)



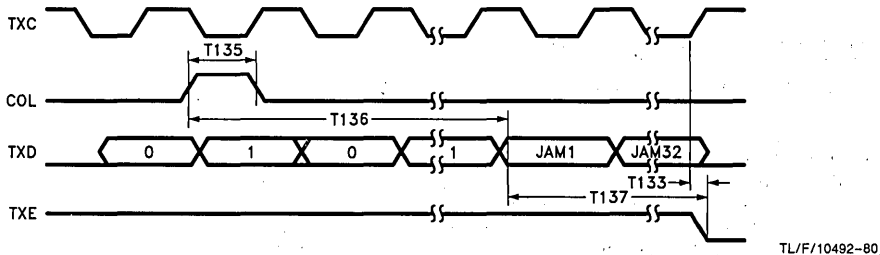
TL/F/10492-79

Number	Parameter	Min	Max	Units
T127	Transmit Clock High Time	40		ns
T128	Transmit Clock Low Time	40		ns
T129	Transmit Clock Cycle Time	90	110	ns
T130	TXC to TXE High		40	ns
T131	TXC to TXD Valid		15	ns
T132	TXD Hold Time from TXC	5		ns
T133	TXC to TXE Low		40	ns
T134	TXE Low to Start of CD Heartbeat (Note 1)		64	$t_{cyc}$
T135	Collision Detect Width (Note 1)	2		$t_{cyc}$

**Note 1:**  $t_{cyc}$  = transmit clock.

## 7.0 AC and DC Specifications (Continued)

### ENDEC-MAC SERIAL TIMING FOR TRANSMISSION (COLLISION)



Number	Parameter	Min	Max	Units
T135	Collision Detect Width (Note 1)	2		tcyc
T136	Delay from Collision		8	tcyc
T137	Jam Period		32	tcyc

Note 1: tcyc = transmit clock.

## 8.0 AC Timing Test Conditions

All specifications are valid only if the mandatory isolation is employed and all differential signals are taken to be at the AUI side of the pulse transformer.

Input Pulse Levels (TTL/CMOS)	GND to 3.0V
Input Rise and Fall Times (TTL/CMOS)	5 ns
Input and Output Reference Levels (TTL/CMOS)	1.5V
Input Pulse Levels (Diff.)	-350 mV to -1315 mV
Input and Output Reference Levels (Diff.)	50% Point of the Differential
TRI-STATE Reference Levels	Float ( $\Delta V$ ) $\pm 0.5V$

## Pin Capacitance

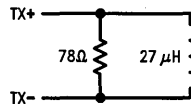
$T_A = 25^\circ C$ ,  $f = 1$  MHz

Symbol	Parameter	Typ	Units
$C_{IN}$	Input Capacitance	7	pF
$C_{OUT}$	Output Capacitance	7	pF

## DERATING FACTOR

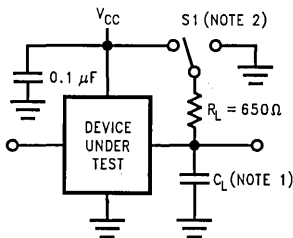
Output timing is measured with a purely capacitive load of 50 pF. The following correction factor can be used for other loads:  $C_L \geq 50$  pF, add 0.05 ns/pF.

## AUI Transmit Test Load



Note: In the above diagram, the TX+ and TX- signals are taken from the AUI side of the isolation (pulse transformer). The pulse transformer used for all testing is a 100  $\mu H \pm 0.1\%$  Pulse Engineering PE64103.

## Output Load (See Figure below)



Note 1: 50 pF, includes scope and jig capacitance.

Note 2: S1 = Open for timing test for push pull outputs.

S1 =  $V_{CC}$  for  $V_{OL}$  test.

S1 = GND for  $V_{OH}$  test.

S1 =  $V_{CC}$  for High Impedance to active low and active low to High Impedance measurements.

S1 = GND for High Impedance to active high and active high to High Impedance measurements.

## DP83916 SONIC™-16 Systems-Oriented Network Interface Controller

### General Description

The SONIC™-16 (Systems-Oriented Network Interface Controller) is a second-generation Ethernet Controller designed to meet the demands of today's high-speed 16-bit systems. Its system interface operates with a high speed DMA that typically consumes less than 8% of the bus bandwidth. Selectable bus modes provide both big and little endian byte ordering and a clean interface to standard microprocessors. The linked-list buffer management system of SONIC-16 offers maximum flexibility in a variety of environments from PC-oriented adapters to high-speed motherboard designs. Furthermore, the SONIC-16 integrates a fully-compatible IEEE 802.3 Encoder/Decoder (ENDEC) allowing for a simple 2-chip solution for Ethernet when the SONIC-16 is paired with the DP8392 Coaxial Transceiver Interface.

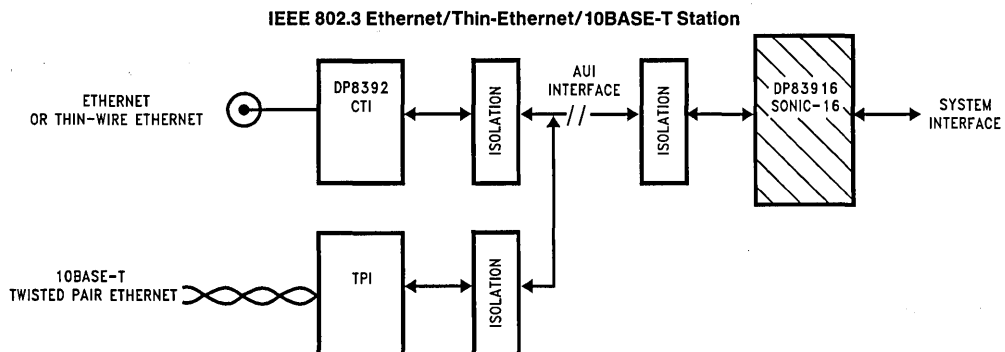
For increased performance, the SONIC-16 implements a unique buffer management scheme to efficiently process receive and transmit packets in system memory. No intermediate packet copy is necessary. The receive buffer management uses three areas in memory for (1) allocating additional resources, (2) indicating status information, and (3) buffering packet data. During reception, the SONIC-16 stores packets in the buffer area, then indicates receive status and control information in the descriptor area. The system allocates more memory resources to the SONIC-16 by adding descriptors to the memory resource area. The transmit buffer management uses two areas in memory:

one for indicating status and control information and the other for fetching packet data. The system can create a transmit queue allowing multiple packets to be transmitted from a single transmit command. The packet data can reside on any arbitrary byte boundary and can exist in several non-contiguous locations.

### Features

- 23-bit non-multiplexed address/16-bit data bus
- High-speed, interruptible DMA
- Linked-list buffer management maximizes flexibility
- Two independent 32-byte transmit and receive FIFOs
- Bus compatibility for all standard microprocessors
- Supports big and little endian formats
- Integrated IEEE 802.3 ENDEC
- Complete address filtering for up to 16 physical and/or multicast addresses
- 32-bit general-purpose timer
- Full-duplex loopback diagnostics
- Fabricated in low-power CMOS
- 132 PQFP package
- Full network management facilities support the IEEE 802.3 layer management standard
- Integrated support for bridge and repeater applications

### System Diagram



TL/F/11722-1

## Table of Contents

### 1.0 FUNCTIONAL DESCRIPTION

- 1.1 IEEE 802.3 ENDEC Unit
  - 1.1.1 ENDEC Operation
  - 1.1.2 Selecting an External ENDEC
- 1.2 MAC Unit
  - 1.2.1 MAC Receive Section
  - 1.2.2 MAC Transmit Section
- 1.3 Byte Ordering
- 1.4 FIFO and Control Logic
  - 1.4.1 Receive FIFO
  - 1.4.2 Transmit FIFO
- 1.5 Status and Configuration Registers
- 1.6 Bus Interface
- 1.7 Loopback and Diagnostics
  - 1.7.1 Loopback Procedure
- 1.8 Network Management Functions

### 2.0 TRANSMIT/RECEIVE IEEE 802.3 FRAME FORMAT

- 2.1 Preamble and Start Of Frame Delimiter (SFD)
- 2.2 Destination Address
- 2.3 Source Address
- 2.4 Length/Type Field
- 2.5 Data Field
- 2.6 FCS Field
- 2.7 MAC (Media Access Control) Conformance

### 3.0 BUFFER MANAGEMENT

- 3.1 Buffer Management Overview
- 3.2 Descriptor Areas
  - 3.2.1 Naming Convention for Descriptors
  - 3.2.2 Abbreviations
  - 3.2.3 Buffer Management Base Address
- 3.3 Descriptor Data Alignment
- 3.4 Receive Buffer Management
  - 3.4.1 Receive Resource Area (RRA)
  - 3.4.2 Receive Buffer Area (RBA)
  - 3.4.3 Receive Descriptor Area (RDA)
  - 3.4.4 Receive Buffer Management Initialization
  - 3.4.5 Beginning of Reception
  - 3.4.6 End of Packet Processing
  - 3.4.7 Overflow Conditions
- 3.5 Transmit Buffer Management
  - 3.5.1 Transmit Descriptor Area (TDA)
  - 3.5.2 Transmit Buffer Area (TBA)
  - 3.5.3 Preparing to Transmit
  - 3.5.4 Dynamically Adding TDA Descriptors

### 4.0 SONIC-16 REGISTERS

- 4.1 The CAM Unit
  - 4.1.1 The Load CAM Command
- 4.2 Status/Control Registers
- 4.3 Register Description
  - 4.3.1 Command Register
  - 4.3.2 Data Configuration Register
  - 4.3.3 Receive Control Register
  - 4.3.4 Transmit Control Register
  - 4.3.5 Interrupt Mask Register
  - 4.3.6 Interrupt Status Register
  - 4.3.7 Data Configuration Register 2
  - 4.3.8 Transmit Registers
  - 4.3.9 Receive Registers
  - 4.3.10 CAM Registers
  - 4.3.11 Tally Counters
  - 4.3.12 General Purpose Timer
  - 4.3.13 Silicon Revision Register

### 5.0 BUS INTERFACE

- 5.1 Pin Configurations
- 5.2 Pin Description
- 5.3 System Configuration
- 5.4 Bus Operations
  - 5.4.1 Acquiring the Bus
  - 5.4.2 Block Transfers
  - 5.4.3 Bus Status
  - 5.4.4 Bus Mode Compatibility
  - 5.4.5 Master Mode Bus Cycles
  - 5.4.6 Bus Exceptions (Bus Retry)
  - 5.4.7 Slave Mode Bus Cycle
  - 5.4.8 On-Chip Memory Arbiter
  - 5.4.9 Chip Reset

### 6.0 NETWORK INTERFACING

- 6.1 Manchester Encoder and Differential Driver
  - 6.1.1 Manchester Decoder
  - 6.1.2 Collision Translator
  - 6.1.3 Oscillator Inputs

### 7.0 AC AND DC SPECIFICATIONS

### 8.0 AC TIMING TEST CONDITIONS

# 1.0 Functional Description

The SONIC-16 (Figure 1-1) consists of an encoder/decoder (ENDEC) unit, media access control (MAC) unit, separate receive and transmit FIFOs, a system buffer management engine, and a user programmable system bus interface unit on a single chip. SONIC-16 is highly pipelined providing maximum system level performance. This section provides a functional overview of SONIC-16.

## 1.1 IEEE 802.3 ENDEC UNIT

The ENDEC (Encoder/Decoder) unit is the interface between the Ethernet transceiver and the MAC unit. It provides the Manchester data encoding and decoding functions for IEEE 802.3 Ethernet/Thin-Ethernet type local area networks. The ENDEC operations of SONIC-16 are identical to the DP83910A CMOS Serial Network Interface device. During transmission, the ENDEC unit combines non-return-zero (NRZ) data from the MAC section and clock pulses into Manchester data and sends the converted data differentially to the transceiver. Conversely, during reception, an analog PLL decodes the Manchester data to NRZ format and receive clock. The ENDEC unit is a functionally complete Manchester encoder/decoder incorporating a balanced driver and receiver, on-board crystal oscillator, collision signal translator, and a diagnostic loopback. The features include:

- Compatible with Ethernet I and II, IEEE 802.3 10BASE5 and 10BASE2
- 10Mb/s Manchester encoding/decoding with receive clock recovery
- Requires no precision components
- Loopback capability for diagnostics
- Externally selectable half or full step modes of operation at transmit output
- Squelch circuitry at the receive and collision inputs reject noise
- Connects to the transceiver (AUI) cable via external pulse transformer

## 1.1.1 ENDEC Operation

The primary function of the ENDEC unit (Figure 1-2) is to perform the encoding and decoding necessary for compatibility between the differential pair Manchester encoded data of the transceiver and the Non-Return-to-Zero (NRZ) serial data of the MAC unit data line. In addition to encoding and decoding the data stream, the ENDEC also supplies all the necessary special signals (e.g., collision detect, carrier sense, and clocks) to the MAC unit.

**Manchester Encoder and Differential Output Driver:** During transmission to the network, the ENDEC unit translates the NRZ serial data from the MAC unit into differential pair Manchester encoded data on the Coaxial Transceiver Interface (e.g., National's DP8392) transmit pair. To perform this operation the NRZ bit stream from the MAC unit is passed through the Manchester encoder block of the ENDEC unit. Once the bit stream is encoded, it is transmitted out differentially to the transmit differential pair through the transmit driver.

**Manchester Decoder:** During reception from the network, the differential receive data from the transceiver (e.g., the DP8392) is converted from Manchester encoded data into NRZ serial data and a receive clock, which are sent to the receive data and clock inputs of the MAC unit. To perform this operation the signal, once received by the differential receiver, is passed to the phase locked loop (PLL) decoder block. The PLL decodes the data and generates a data receive clock and a NRZ serial data stream to the MAC unit.

**Special Signals:** In addition to performing the Manchester encoding and decoding function, the ENDEC unit provides control and clocking signals to the MAC unit. The ENDEC sends a carrier sense (CRS) signal that indicates to the MAC unit that data is present on the network on the ENDEC's receive differential pair. The MAC unit is also provided with a collision detection signal (COL) that informs the MAC unit that a collision is taking place somewhere on the

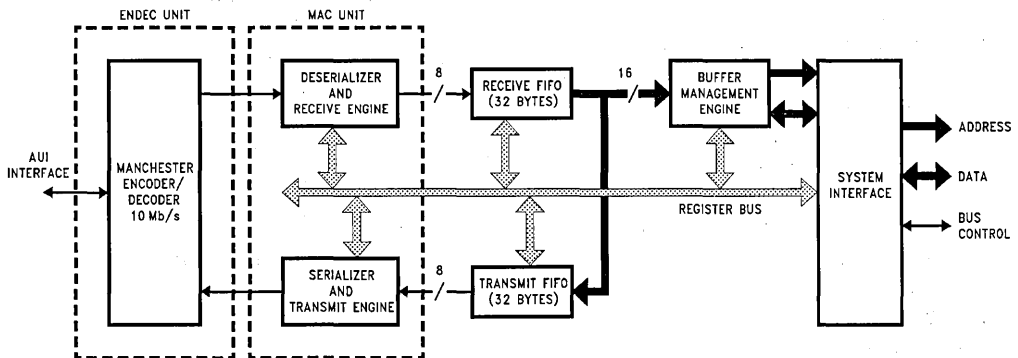


FIGURE 1-1. SONIC-16 Block Diagram

TL/F/11722-2

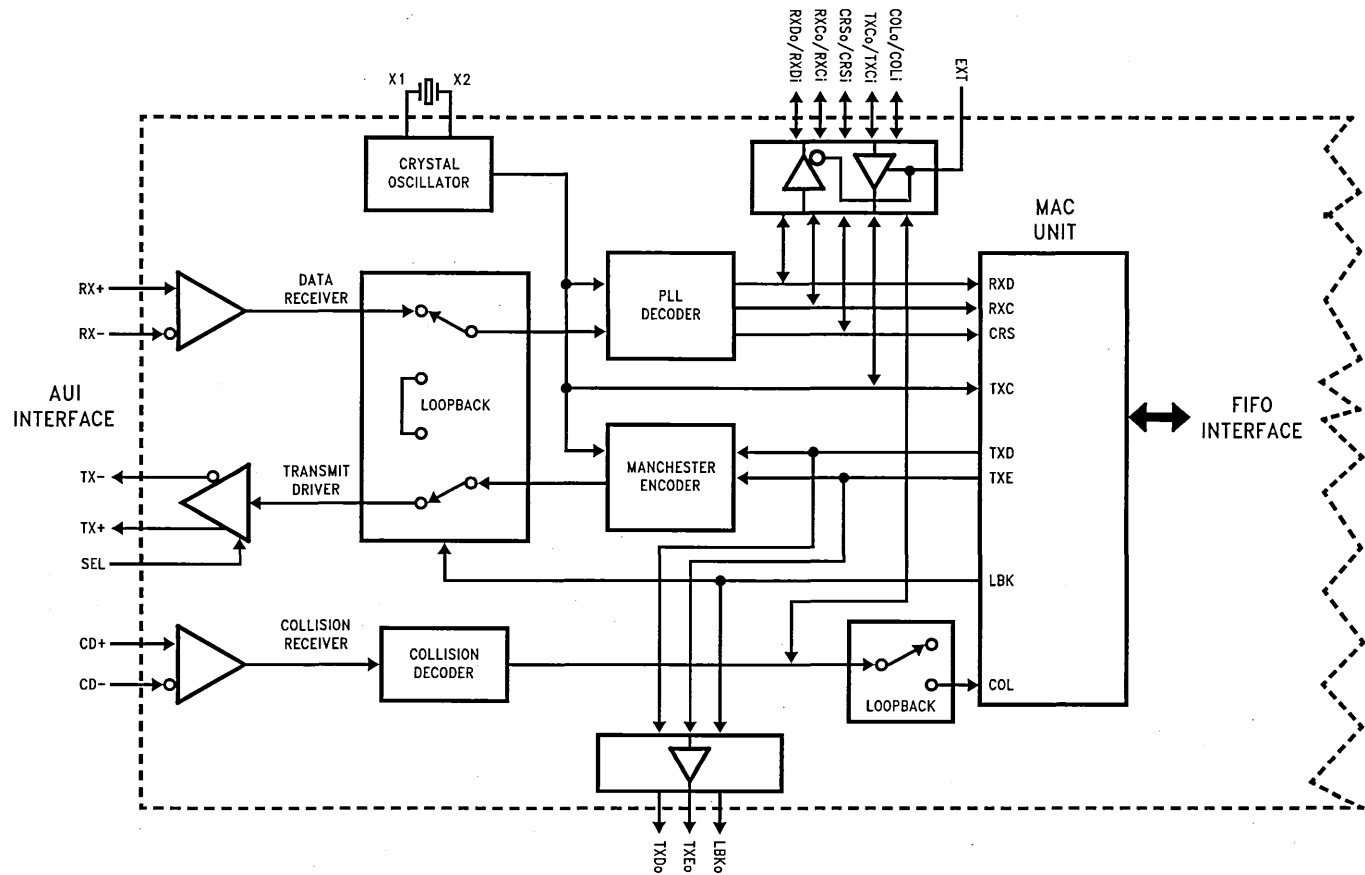


FIGURE 1-2. Block Diagram of Ethernet ENDEC

TL/F/11722-3

1-737

## 1.0 Functional Description (Continued)

network. The ENDEC section detects this when its collision receiver detects a 10 MHz signal on the differential collision input pair. The ENDEC also provides both the receive and transmit clocks to the MAC unit. The transmit clock is one half of the oscillator input. The receive clock is extracted from the input data by the PLL.

**Oscillator:** The oscillator generates the 10 MHz transmit clock signal for network timing. The oscillator is controlled by a parallel resonant crystal or by an external clock (see section 6.1.3). The 20 MHz output of the oscillator is divided by 2 to generate the 10 MHz transmit clock (TXC) for the MAC section. The oscillator provides an internal clock signal for the encoding and decoding circuits.

The signals provided to the MAC unit from the on-chip ENDEC are also provided as outputs to the user.

**Loopback Functions:** The SONIC-16 provides three loopback modes. These modes allow loopback testing at the MAC, ENDEC and external transceiver level (see section 1.7 for details). It is important to note that when the SONIC-16 is transmitting, the transmitted packet will always be looped back by the external transceiver. The SONIC-16 takes advantage of this to monitor the transmitted packet. See the explanation of the Receive State Machine in section 1.2.1 for more information about monitoring transmitted packets.

### 1.1.2 Selecting An External ENDEC

An option is provided on SONIC-16 to disable the on-chip ENDEC unit and use an external ENDEC. The internal IEEE 802.3 ENDEC can be bypassed by connecting the EXT pin to V<sub>CC</sub> (EXT = 1). In this mode the MAC signals are redirected out from the chip, allowing an external ENDEC to be used. See section 5.2 for the alternate pin definitions.

### 1.2 MAC UNIT

The MAC (Media Access Control) unit performs the media access control functions for transmitting and receiving packets over Ethernet. During transmission, the MAC unit frames information from the transmit FIFO and supplies serialized data to the ENDEC unit. During reception, the incoming information from the ENDEC unit is deserialized, the frame checked for valid reception, and the data is transferred to the receive FIFO. Control and status registers on the SONIC-16 govern the operation of the MAC unit.

#### 1.2.1 MAC Receive Section

The receive section (*Figure 1-3*) controls the MAC receive operations during reception, loopback, and transmission. During reception, the deserializer goes active after detecting the 2-bit SFD (Start of Frame Delimiter) pattern (section 2.1). It then frames the incoming bits into octet boundaries

and transfers the data to the 32-byte receive FIFO. Concurrently the address comparator compares the Destination Address Field to the addresses stored in the chip's CAM address registers (Content Addressable Memory cells). If a match occurs, the deserializer passes the remainder of the packet to the receive FIFO. The packet is decapsulated when the carrier sense input pin (CRS) goes inactive. At the end of reception the receive section checks the following:

- Frame alignment errors
- CRC errors
- Length errors (runt packets)

The appropriate status is indicated in the Receive Control register (section 4.3.3). In loopback operations, the receive section operates the same as during normal reception.

During transmission, the receive section remains active to allow monitoring of the self-received packet. The CRC checker operates as normal, and the Source Address field is compared with the CAM address entries. Status of the CRC check and the source address comparison is indicated by the PMB bit in the Transmit Control register (section 4.3.4). No data is written to the receive FIFO during transmit operations.

The receive section consists of the following blocks detailed below.

**Receive State Machine (RSM):** The RSM insures the proper sequencing for normal reception and self-reception during transmission. When the network is inactive, the RSM remains in an idle state continually monitoring for network activity. If the network becomes active, the RSM allows the deserializer to write data into the receive FIFO. During this state, the following conditions may prevent the complete reception of the packet.

- FIFO Overflow—The receive FIFO has been completely filled before the SONIC-16 could buffer the data to memory.
- CAM Address Mismatch—The packet is rejected because of a mismatch between the destination address of the packet and the address in the CAM.
- Memory Resource Error—There are no more resources (buffers) available for buffering the incoming packets.
- Collision or Other Error—A collision occurred on the network or some other error, such as a CRC error, occurred (this is true if the SONIC-16 has been told to reject packets on a collision, or reject packets with errors).

If these conditions do not occur, the RSM processes the packet indicating the appropriate status in the Receive Control register.

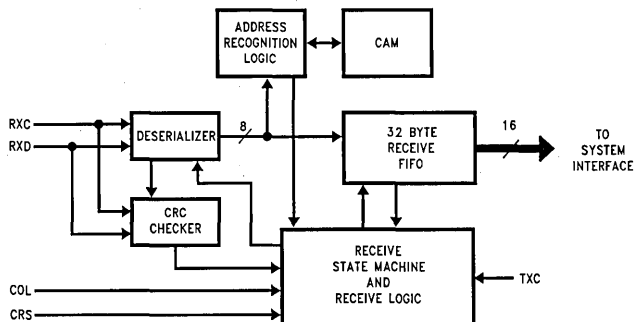


FIGURE 1-3. MAC Receiver

TL/F/11722-4



## 1.0 Functional Description (Continued)

During transmission of a packet from the SONIC-16, the external transceiver will always loop the packet back to the SONIC-16. The SONIC-16 will use this to monitor the packet as it is being transmitted. The CRC and source address of the looped back packet are checked with the CRC and source address that were transmitted. If they do not match, an error bit is set in the status of the transmitted packet (see Packet Monitored Bad, PBM, in the Transmit Control Register, section 4.3.4). Data is not written to the receive FIFO during this monitoring process unless Transceiver Loopback mode has been selected (see section 1.7).

**Receive Logic:** The receive logic contains the command, control, and status registers that govern the operations of the receive section. It generates the control signals for writing data to the receive FIFO, processes error signals obtained from the CRC checker and the deserializer, activates the "packet reject" signal to the RSM for rejecting packets, and posts the applicable status in the Receive Control register.

**Deserializer:** This section deserializes the serial input data stream and furnishes a byte clock for the address comparator and receive logic. It also synchronizes the CRC checker to begin operation (after SFD is detected), and checks for proper frame alignment with respect to CRS going inactive at the end of reception.

**Address Comparator:** The address comparator latches the Destination Address (during reception or loopback) or Source Address (during transmission) and determines whether the address matches one of the entries in the CAM (Content Addressable Memory).

**CRC Checker:** The CRC checker calculates the 4-byte Frame Check Sequence (FCS) field from the incoming data stream and compares it with the last 4-bytes of the received packet. The CRC checker is active for both normal reception and self-reception during transmission.

**Content Addressable Memory (CAM):** The CAM contains 16 user programmable entries and 1 pre-programmed Broadcast address entry for complete filtering of received packets. The CAM can be loaded with any combination of Physical and Multicast Addresses (section 2.2). See section 4.1 for the procedure on loading the CAM registers.

### 1.2.2 MAC Transmit Section

The transmit section (Figure 1-4) is responsible for reading data from the transmit FIFO and transmitting a serial data

stream onto the network in conformance with the IEEE 802.3 CSMA/CD standard. The Transmit Section consists of the following blocks.

**Transmit State Machine (TSM):** The TSM controls the functions of the serializer, preamble generator, and JAM generator. It determines the proper sequence of events that the transmitter follows under various network conditions. If no collision occurs, the transmitter prefixes a 62-bit preamble and 2-bit Start of Frame Delimiter (SFD) at the beginning of each packet, then sends the serialized data. At the end of the packet, an optional 4-byte CRC pattern is appended. If a collision occurs, the transmitter switches from transmitting data to sending a 4-byte Jam pattern to notify all nodes that a collision has occurred. Should the collision occur during the preamble, the transmitter waits for it to complete before jamming. After the transmission has completed, the transmitter writes status in the Transmit Control register (section 4.3.4).

**Protocol State Machine:** The protocol state machine assures that the SONIC-16 obeys the CSMA/CD protocol. Before transmitting, this state machine monitors the carrier sense and collision signals for network activity. If another node(s) is currently transmitting, the SONIC-16 defers until the network is quiet, then transmits after its Interframe Gap Timer (9.6  $\mu$ s) has expired. The Interframe Gap time is divided into two portions. During the first 6.4  $\mu$ s, network activity restarts the Interframe Gap timer. Beyond this time, however, network activity is ignored and the state machine waits the remaining 3.2  $\mu$ s before transmitting. If the SONIC-16 experiences a collision during a transmission, the SONIC-16 switches from transmitting data to a 4-byte JAM pattern (4 bytes of all 1's), before ceasing to transmit. The SONIC-16 then waits a random number of slot times (51.2  $\mu$ s) determined by the *Truncated Binary Exponential Backoff Algorithm* before reattempting another transmission. In this algorithm, the number of slot times to delay before the *n*th retransmission is chosen to be a random integer *r* in the range of:

$$0 \leq r \leq 2^k$$

$$\text{where } k = \min(n, 10)$$

If a collision occurs on the 16th transmit attempt, the SONIC-16 aborts transmitting the packet and reports an "Excessive Collisions" error in the Transmit Control register.

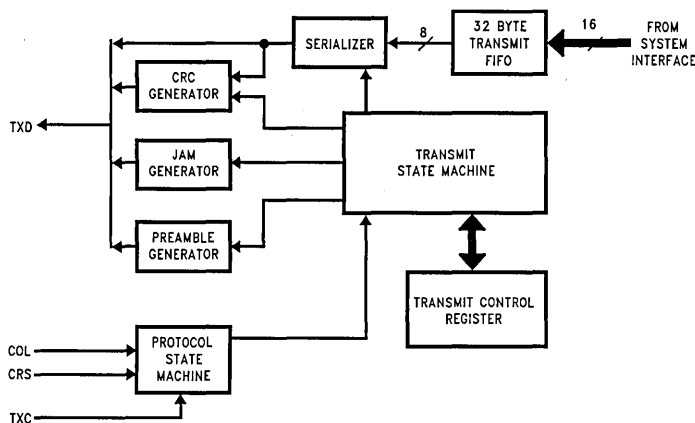


FIGURE 1-4. MAC Transmitter

TL/F/11722-5

### 1.0 Functional Description (Continued)

**Serializer:** After data has been written into the 32-byte transmit FIFO, the serializer reads byte wide data from the FIFO and sends a NRZ data stream to the Manchester encoder. The rate at which data is transmitted is determined by the transmit clock (TXC). The serialized data is transmitted after the SFD.

**Preamble Generator:** The preamble generator prefixes a 62-bit alternating "1,0" pattern and a 2-bit "1,1" SFD pattern at the beginning of each packet. This allows receiving nodes to synchronize to the incoming data. The preamble is always transmitted in its entirety even in the event of a collision. This assures that the minimum collision fragment is 96 bits (64 bits of normal preamble, and 4 bytes, or rather 32 bits, of the JAM pattern).

**CRC Generator:** The CRC generator calculates the 4-byte FCS field from the transmitted serial data stream. If enabled, the 4-byte FCS field is appended to the end of the transmitted packet (section 2.6).

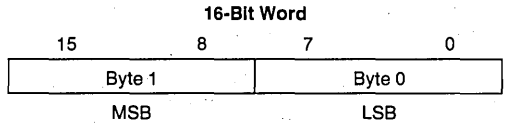
**Jam Generator:** The Jam generator produces a 4-byte pattern of all 1's to assure that all nodes on the network sense the collision. When a collision occurs, the SONIC-16 stops transmitting data and enables the Jam generator. If a collision occurs during the preamble, the SONIC-16 finishes transmitting the preamble before enabling the Jam generator (see Preamble Generator above).

#### 1.3 BYTE ORDERING

The SONIC-16 will operate with 16-bit wide memory. The SONIC-16 provides both Little Endian and Big Endian byte-

ordering capability for compatibility with National/Intel or Motorola microprocessors respectively by selecting the proper level on the BMODE pin. The byte ordering is depicted as follows:

**Little Endian mode (BMODE = 0):** The byte orientation for received and transmitted data in the Receive Buffer Area (RBA) and Transmit Buffer Area (TBA) of system memory is as follows:



**Big Endian mode (BMODE = 1):** The byte orientation for received and transmitted data in the RBA and TBA is as follows:

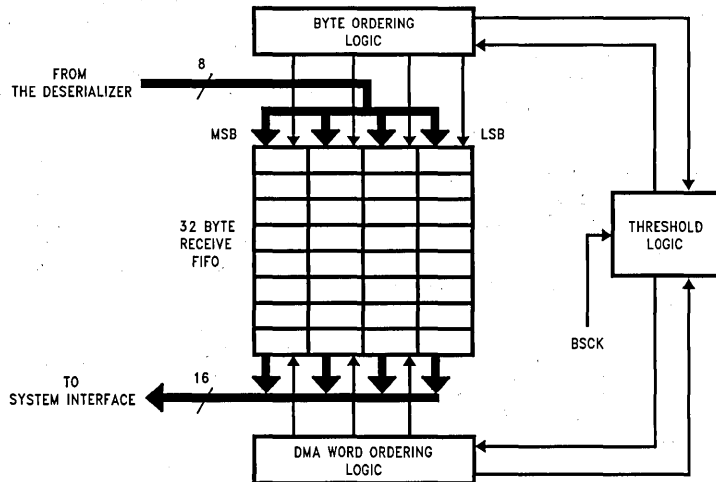
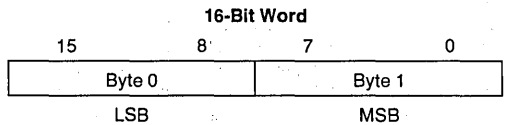


FIGURE 1-5. Receive FIFO

TL/F/11722-6

## 1.0 Functional Description (Continued)

### 1.4 FIFO AND CONTROL LOGIC

The SONIC-16 incorporates two independent 32-byte FIFOs for transferring data to/from the system interface and from/to the network. The FIFOs, providing temporary storage of data, free the host system from the real-time demands on the network.

The way in which the FIFOs are emptied and filled is controlled by the FIFO threshold values and the Block Mode Select bits (BMS, section 4.3.2). The threshold values determine how full or empty the FIFOs can be before the SONIC-16 will request the bus to get more data from memory or buffer more data to memory. When block mode is set, the number of bytes transferred is set by the threshold value. For example, if the threshold for the receive FIFO is 4 words, then the SONIC-16 will always transfer 4 words from the receive FIFO to memory. If empty/fill mode is set, however, the number of bytes transferred is the number required to fill the transmit FIFO or empty the receive FIFO. More specific information about how the threshold affects reception and transmission of packets is discussed in sections 1.4.1 and 1.4.2 below.

#### 1.4.1 Receive FIFO

To accommodate the different transfer rates, the receive FIFO (*Figure 1-5*) serves as a buffer between the 8-bit network (deserializer) interface and the 16-bit system interface. The FIFO is arranged as a 4-byte wide by 8 deep memory array (8 long words, or 32 bytes) controlled by three sections of logic. During reception, the Byte Ordering logic directs the byte stream from the deserializer into the FIFO using one of four write pointers. Depending on the selected byte-ordering mode, data is written either least significant byte first or most significant byte first to accommodate little or big endian byte-ordering formats respectively.

As data enters the FIFO, the Threshold Logic monitors the number of bytes written in from the deserializer. The programmable threshold (RFT1,0 in the Data Configuration Register) determines the number of words (or long words) written into the FIFO from the MAC unit before a DMA request for system memory occurs. When the threshold is reached, the Threshold Logic enables the Buffer Management Engine to read a programmed number of 16-bit words (depending upon the selected word width) from the FIFO and transfers them to the system interface (the system memory) using DMA. The threshold is reached when the number of bytes in the receive FIFO is greater than the value of the threshold. For example, if the threshold is 4 words (8 bytes), then the Threshold Logic will not cause the Buffer Management Engine to write to memory until there are more than 8 bytes in the FIFO.

The Buffer Management Engine reads either the upper or lower half (16 bits) of the FIFO. If, after the transfer is complete, the number of bytes in the FIFO is less than the threshold, then the SONIC-16 is done. This is always the case when the SONIC-16 is in empty/fill mode. If, however, for some reason (e.g. latency on the bus) the number of bytes in the FIFO is still greater than the threshold value, the Threshold Logic will cause the Buffer Management Engine to do a DMA request to write to memory again. This later case is usually only possible when the SONIC-16 is in block mode.

When in block mode, each time the SONIC-16 requests the bus, only a number of bytes equal to the threshold value will

be transferred. The Threshold Logic continues to monitor the number of bytes written in from the deserializer and enables the Buffer Management Engine every time the threshold has been reached. This process continues until the end of the packet.

Once the end of the packet has been reached, the serializer will fill out the last word if the last byte did not end on a word boundary. The fill byte will be 0FFh. Immediately after the last byte (or fill byte) in the FIFO, the received packets status will be written into the FIFO. The entire packet, including any fill bytes and the received packet status will be buffered to memory. When a packet is buffered to memory by the Buffer Management Engine, it is always taken from the FIFO in words and buffered to memory on word boundaries. Data from a packet cannot be buffered on odd byte boundaries (see Section 3.3). For more information on the receive packet buffering process, see Section 3.4.

#### 1.4.2 Transmit FIFO

Similar to the Receive FIFO, the Transmit FIFO (*Figure 1-6*) serves as a buffer between the 16-bit system interface and the network (serializer) interface. The Transmit FIFO is also arranged as a 4 byte by 8 deep memory array (8 long words or 32 bytes) controlled by three sections of logic. Before transmission can begin, the Buffer Management Engine fetches a programmed number of 16-bit words from memory and transfers them to the FIFO. The Buffer Management Engine writes either the upper or lower half (16 bits) into the FIFO.

The Threshold logic monitors the number of bytes as they are written into the FIFO. When the threshold has been reached, the Transmit Byte Ordering state machine begins reading bytes from the FIFO to produce a continuous byte stream for the serializer. The threshold is met when the number of bytes in the FIFO is greater than the value of the threshold. For example, if the transmit threshold is 4 words (8 bytes), the Transmit Byte Ordering state machine will not begin reading bytes from the FIFO until there are 9 or more bytes in the buffer. The Buffer Management Engine continues replenishing the FIFO until the end of the packet. It does this by making multiple DMA requests to the system interface. Whenever the number of bytes in the FIFO is equal to or less than the threshold value, the Buffer Management Engine will do a DMA request. If block mode is set, then after each request has been granted by the system, the Buffer Management Engine will transfer a number of bytes equal to the threshold value into the FIFO. If empty/fill mode is set, the FIFO will be completely filled in one DMA request.

Since data may be organized in big or little endian byte ordering format, the Transmit Byte Ordering state machine uses one of four read pointers to locate the proper byte within the 4 byte wide FIFO. It also determines the valid number of bytes in the FIFO. For packets which begin or end at odd bytes in the FIFO, the Buffer Management Engine writes extraneous bytes into the FIFO. The Transmit Byte Ordering state machine detects these bytes and only transfers the valid bytes to the serializer. The Buffer Management Engine can read data from memory on any byte boundary (see Section 3.3). See Section 3.5 for more information on transmit buffering.

## 1.0 Functional Description (Continued)

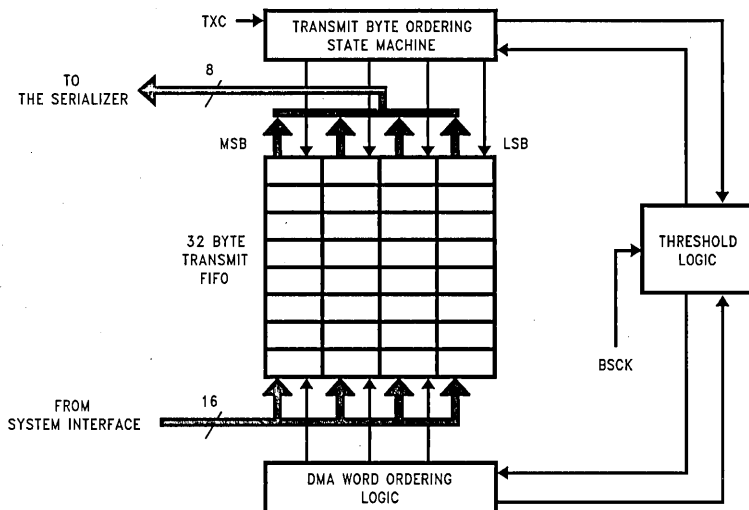


FIGURE 1-6. Transmit FIFO

TL/F/11722-7

### 1.5 STATUS AND CONFIGURATION REGISTERS

The SONIC-16 contains a set of status/control registers for conveying status and control information to/from the host system. The SONIC-16 uses these registers for loading commands generated from the system, indicating transmit and receive status, buffering data to/from memory, and providing interrupt control. Each register is 16 bits in length. See section 4.0 for a description of the registers.

### 1.6 BUS INTERFACE

The system interface (Figure 1-7) consists of the pins necessary for interfacing to a variety of buses. It includes the I/O drivers for the data and address lines, bus access control for standard microprocessors, ready logic for synchronous or asynchronous systems, slave access control, interrupt control, and shared-memory access control. The functional signal groups are shown in Figure 1-7. See section 5.0 for a complete description of the SONIC-16 bus interface.

### 1.7 LOOPBACK AND DIAGNOSTICS

The SONIC-16 furnishes three loopback modes for self-testing from the controller interface to the transceiver interface. The loopback function is provided to allow self-testing of the chip's internal transmit and receive operations. During loopback, transmitted packets are routed back to the receive section of the SONIC-16 where they are filtered by the address recognition logic and buffered to memory if accepted. Transmit and receive status and interrupts remain active during loopback. This means that when using loopback, it is as if the packet was transmitted and received by two separate chips that are connected to the same bus and memory.

**MAC Loopback:** Transmitted data is looped back at the MAC. Data is not sent from the MAC to either the internal ENDEC or an external ENDEC (the external ENDEC interface pins will not be driven), hence, data is not transmitted from the chip. Even though the ENDEC is not used in MAC loopback, the ENDEC clock (an oscillator or crystal for the internal ENDEC or TXC for an external ENDEC) must be driven. Network activity, such as a collision, does not affect

MAC loopback. CSMA/CD MAC protocol is not completely followed in MAC loopback.

**ENDEC Loopback:** Transmitted data is looped back at the ENDEC. If the internal ENDEC is used, data is switched from the transmit section of the ENDEC to the receive section (Figure 1-2). Data is not transmitted from the chip and the collision lines,  $CD\pm$ , are ignored, hence, network activity does not affect ENDEC loopback. The LBK signal from the MAC tells the internal ENDEC to go into loopback mode. If an external ENDEC is used, it should operate in loopback mode when the LBK signal is asserted. CSMA/CD MAC protocol is followed even though data is not transmitted from the chip.

**Transceiver Loopback:** Transmitted data is looped back at the external transceiver (which is always the case regardless of the SONIC-16's loopback mode). CSMA/CD MAC protocol is followed since data will be transmitted from the chip. This means that transceiver loopback is affected by network activity. The basic difference between Transceiver Loopback and normal, non-loopback, operations of the SONIC-16 is that in Transceiver Loopback, the SONIC-16 loads the receive FIFO and buffers the packet to memory. In normal operations, the SONIC-16 only monitors the packet that is looped back by the transceiver, but does not fill the receive FIFO and buffer the packet.

#### 1.7.1 Loopback Procedure

The following procedure describes the loopback operation.

1. Initialize the Transmit and Receive Area as described in Sections 3.4 and 3.5.
2. Load one of the CAM address registers (see Section 4.1), with the Destination Address of the packet if you are verifying the SONIC-16's address recognition capability.
3. Load one of the CAM address registers with the Source Address of the packet if it is different than the Destination Address to avoid getting a Packet Monitored Bad (PMB) error in the Transmit status (see Section 4.3.4).

### 1.0 Functional Description (Continued)

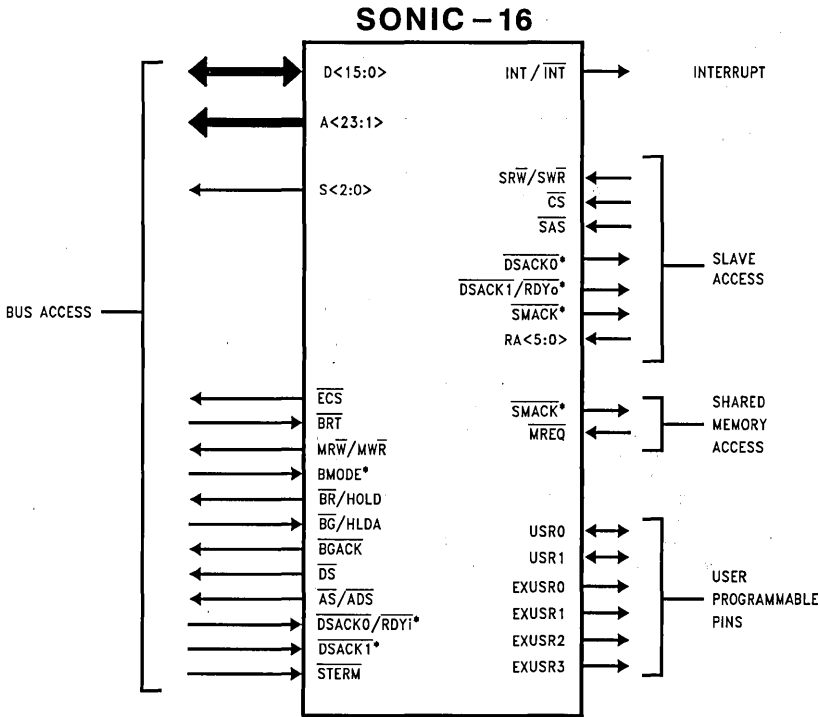
4. Program the Receive Control register with the desired receive filter and the loopback mode (LB1, LB0).
5. Issue the transmit command (TXP) and enable the receiver (RXEN) in the Command register.

The SONIC-16 completes the loopback operation after the packet has been completely received (or rejected if there is an address mismatch). The Transmit Control and Receive Control registers treat the loopback packet as in normal operation and indicate status accordingly. Interrupts are also generated if enabled in the Interrupt Mask register.

**Note:** For MAC Loopback, only one packet may be queued for proper operation. This restriction occurs because the transmit MAC section, which does not generate an Interframe Gap time (IFG) between transmitted packets, does not allow the receive MAC section to update receive status. There are no restrictions for the other loopback modes.

### 1.8 NETWORK MANAGEMENT FUNCTIONS

The SONIC-16 fully supports the Layer Management IEEE 802.3 standard to allow a node to monitor the overall performance of the network. These statistics are available on a per packet basis at the end of reception or transmission. In addition, the SONIC-16 provides three tally counters to tabulate CRC errors, Frame Alignment errors, and missed packets. Table 1-1 shows the statistics indicated by the SONIC-16.



TL/F/11722-8

\*Note: DSACK0,T are used for both Bus and Slave Access Control and are bidirectional. SMACK is used for both Slave access and shared memory access. The BMODE pin selects between National/Intel or Motorola type buses.

**FIGURE 1-7. SONIC-16 Interface Signals**

# 1.0 Functional Description (Continued)

TABLE 1-1. Network Management Statistics

Statistic	Register Used	Bits Used
Frames Transmitted OK	TCR (Note)	PTX
Single Collision Frames	(Note)	NC0-NC4
Multiple Collision Frames	(Note)	NC0-NC4
Collision Frames	(Note)	NC0-NC4
Frames with Deferred Transmissions	TCR (Note)	DEF
Late Collisions	TCR (Note)	OWC
Excessive Collisions	TCR (Note)	EXC
Excessive Deferral	TCR (Note)	EXD
Internal MAC Transmit Error	TCR (Note)	BCM, FU
Frames Received OK	RCR (Note)	PRX
Multicast Frames Received OK	RCR (Note)	MC
Broadcast Frames Received OK	RCR (Note)	BC
Frame Check Sequence Errors	CRCT RCR	All CRC
Alignment Errors	FAET RCR	All FAE
Frame Lost due to Internal MAC Receive Error	MPT ISR	All RFO

Note: The number of collisions and the contents of the Transmit Control register are posted in the TXpkt.status field (see section 3.5.1.2). The contents of the Receive Control register are posted in the RXpkt.status field (see section 3.4.3.1).

## 2.0 Transmit/Receive IEEE 802.3 Frame Format

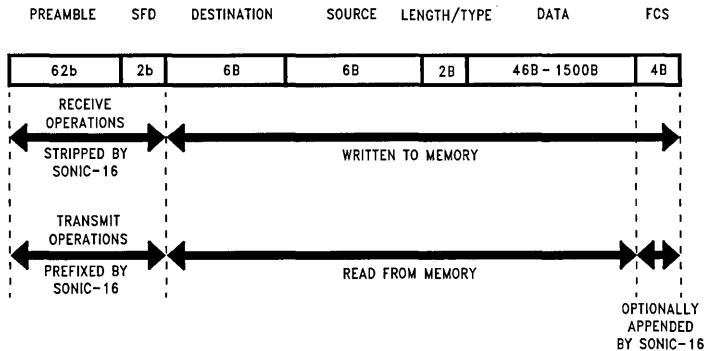
A standard IEEE 802.3 packet (Figure 2-1) consists of the following fields: preamble, Start of Frame Delimiter (SFD), destination address, source address, length, data and Frame Check Sequence (FCS). The typical format is shown in Figure 2-1. The packets are Manchester encoded and decoded by the ENDEC unit and transferred serially to/from the MAC unit using NRZ data with a clock. All fields are of fixed length except for the data field. The SONIC-16 generates and appends the preamble, SFD and FCS field during transmission. The Preamble and SFD fields are stripped during reception. (The CRC is passed through to buffer memory during reception.)

### 2.1 PREAMBLE AND START OF FRAME DELIMITER (SFD)

The Manchester encoded alternating 1,0 preamble field is used by the ENDEC to acquire bit synchronization with an incoming packet. When transmitted, each packet contains 62 bits of an alternating 1,0 preamble. Some of this preamble may be lost as the packet travels through the network. Byte alignment is performed when the Start of Frame Delimiter (SFD) pattern, consisting of two consecutive 1's, is detected.

### 2.2 DESTINATION ADDRESS

The destination address indicates the destination of the packet on the network and is used to filter unwanted pack-



Note: B = bytes  
b = bits

FIGURE 2-1. IEEE 802.3 Packet Structure

TL/F/11722-9

## 2.0 Transmit/Receive IEEE 802.3 Frame Format (Continued)

ets from reaching a node. There are three types of address formats supported by the SONIC-16: Physical, Multicast, and Broadcast.

**Physical Address:** The physical address is a unique address that corresponds only to a single node. All physical addresses have the LSB of the first byte of the address set to "0". These addresses are compared to the internally stored CAM (Content Addressable Memory) address entries. All bits in the destination address must match an entry in the CAM in order for the SONIC-16 to accept the packet.

**Multicast Address:** Multicast addresses, which have the LSB of the first byte of the address set to "1", are treated similarly as Physical addresses, i.e., they must match an entry in the CAM. This allows perfect filtering of Multicast packet's and eliminates the need for a hashing algorithm for mapping Multicast packets.

**Broadcast Address:** If the address consists of all 1's, it is a Broadcast address, indicating that the packet is intended for all nodes.

The SONIC-16 also provides a promiscuous mode which allows reception of all physical address packets. Physical, Multicast, Broadcast, and promiscuous address modes can be selected via the Receive Control register.

### 2.3 SOURCE ADDRESS

The source address is the physical address of the sending node. Source addresses cannot be multicast or broadcast addresses. This field must be passed to the SONIC-16's transmit buffer from the system software. During transmission, the SONIC-16 compares the Source address with its internal CAM address entries before monitoring the CRC of the self-received packet. If the source address of the packet transmitted does not match a value in the CAM, the packet monitored bad flag (PMB) will be set in the transmit status field of the transmit descriptor (see Sections 3.5.1.2 and 4.3.4). The SONIC-16 does not provide Source Address insertion. However, a transmit descriptor fragment, containing only the Source Address, may be created for each packet. (See Section 3.5.1.)

### 2.4 LENGTH/TYPE FIELD

For IEEE 802.3 type packets, this field indicates the number of bytes that are contained in the data field of the packet. For Ethernet I and II networks, this field indicates the type of packet. The SONIC-16 does not operate on this field.

### 2.5 DATA FIELD

The data field has a variable octet length ranging from 46 to 1500 bytes as defined by the Ethernet specification. Messages longer than 1500 bytes need to be broken into multiple packets for IEEE 802.3 networks. Data fields shorter than 46 bytes require appending a pad to bring the complete frame length to 64 bytes. If the data field is padded, the number of valid bytes are indicated in the length field. The SONIC-16 does not append pad bytes for short packets during transmission, nor check for oversize packets during reception. However, the user's driver software can easily append the pad by lengthening the TXpkt.pkt\_size field and TXpkt.frag\_size field(s) to at least 64 bytes (see Section 3.5.1). While the Ethernet specification defines the maximum number of bytes in the data field the SONIC-16 can transmit and receive packets up to 64k bytes.

### 2.6 FCS FIELD

The Frame Check Sequence (FCS) is a 32-bit CRC field calculated and appended to a packet during transmission to allow detection of error-free packets. During reception, an error-free packet results in a specific pattern in the CRC

generator. The AUTODIN II ( $X_{32} + X_{26} + X_{23} + X_{22} + X_{16} + X_{12} + X_{11} + X_{10} + X_8 + X_7 + X_5 + X_4 + X_2 + X_1 + 1$ ) polynomial is used for the CRC calculations. The SONIC-16 may optionally append the CRC sequence during transmission, and checks the CRC both during normal reception and self-reception during a transmission (see Section 1.2.1).

### 2.7 MAC (MEDIA ACCESS CONTROL) CONFORMANCE

The SONIC-16 is designed to be compliant to the IEEE 802.3 MAC Conformance specification. The SONIC-16 implements most of the MAC functions in silicon and provides hooks for the user software to handle the remaining functions. The MAC Conformance specifications are summarized in Table 2-1.

TABLE 2-1. MAC Conformance Specifications

Conformance Test Name	Support By		
	SONIC-16	User Driver Software	Notes
Minimum Frame Size	X		
Maximum Frame Size	X	X	1
Address Generation	X	X	2
Address Recognition	X		
Pad Length Generation	X	X	3
Start Of Frame Delimiter	X		
Length Field	X		
Preamble Generation	X		
Order of Bit Transmission	X		
Inconsistent Frame Length	X	X	1
Non-Integral Octet Count	X		
Incorrect Frame Check Sequence	X		
Frame Assembly	X		
FCS Generation and Insertion	X		
Carrier Deference	X		
Interframe Spacing	X		
Collision Detection	X		
Collision Handling	X		
Collision Backoff and Retransmission	X		
FCS Validation	X		
Frame Disassembly	X		
Back-to-Back Frames	X		
Flow Control	X		
Attempt Limit	X		
Jam Size (after SFD)	X		
Jam Size (in Preamble)	X		

**Note 1:** The SONIC-16 provides the byte count of the entire packet in the RXpkt.byte\_count (see Section 3.4.3). The user's driver software may perform further filtering of the packet based upon the byte count.

**Note 2:** The SONIC-16 does not provide Source Address insertion; however, a transmit descriptor fragment, containing only the Source Address, may be created for each packet. See Section 3.5.1.

**Note 3:** The SONIC-16 does not provide Pad generation; however, the user's driver software can easily append the Pad by lengthening the TXpkt.pkt\_size field and TXpkt.frag\_size field(s) to at least 64 bytes. See Section 3.5.1.

### 3.0 Buffer Management

#### 3.1 BUFFER MANAGEMENT OVERVIEW

The SONIC-16's buffer management scheme is based on separate buffers and descriptors (*Figures 3-2 and 3-11*). Packets that are received or transmitted are placed in buffers called the Receive Buffer Area (RBA) and the Transmit Buffer Area (TBA). The system keeps track of packets in these buffers using the information in the Receive Descriptor Area (RDA) and the Transmit Descriptor Area (TDA). A single (TDA) points to a single TBA, but multiple RDAs can point to a single RBA (one RDA per packet in the buffer). The Receive Resource Area (RRA), which is another form of descriptor, is used to keep track of the actual buffer.

When packets are transmitted, the system sets up the packets in one or more TBAs with a TDA pointing to each TBA. There can only be one packet per TBA/TDA pair. A single packet, however, may be made up of several fragments of data dispersed in memory. There is one TDA pointing to each packet which specifies information about the packet's size, location in memory, number of fragments and status after transmission. The TDAs are linked together in a linked list. The system causes the SONIC-16 to transmit the packets by passing the first TDA to the SONIC-16 and issuing the transmit command.

Before a packet can be received, an RBA and RDA must be set up by the system. RDAs are made up as a linked list similar to TDAs. An RDA is not linked to a particular RBA, though. Instead, an RDA is linked specifically to a packet after it has been buffered into an RBA. More than one packet can be buffered into the same RBA, but each packet gets its own RDA. A received packet can not be scattered into fragments. The system only needs to tell the SONIC-16 where the first RDA and where the RBAs are. Since an RDA never specifically points to an RBA, the RRA is used to keep track of the RBAs. The RRA is a circular queue of pointers and buffer sizes (not a linked list). When the SONIC-16 receives a packet, it is buffered into a RBA and a RDA is written to so that it points to and describes the new packet. If the RBA does not have enough space to buffer the next packet, a new RBA is obtained from the RRA.

#### 3.2 DESCRIPTOR AREAS

Descriptors are the basis of the buffer management scheme used by the SONIC-16. A RDA points to a received packet within a RBA, a RRA points to a RBA and a TDA points to a TBA which contains a packet to be transmitted. The conventions and registers used to describe these descriptors are discussed in the next three sections.

##### 3.2.1 Naming Convention for Descriptors

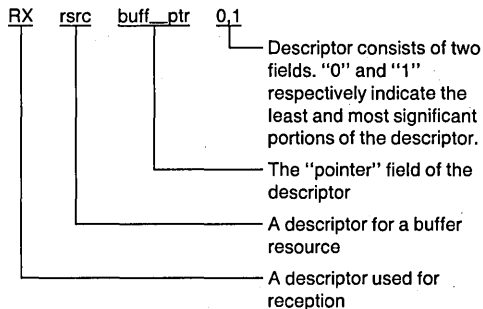
The fields which make up the descriptors are named in a consistent manner to assist in remembering the usage of each descriptor. Each descriptor name consists of three components in the following format.

[RX/TX][descriptor name].[field]

The first two capital letters indicate whether the descriptor is used for transmission (TX) or reception (RX), and is then followed by the descriptor name having one of two names.

rsrc = Resource descriptor  
 pkt = Packet descriptor

The last component consists of a field name to distinguish it from the other fields of a descriptor. The field name is separated from the descriptor name by a period ("."). An example of a descriptor is shown below.



##### 3.2.2 Abbreviations

The abbreviations in Table 3-1 are used to describe the SONIC-16 registers and data structures in memory. The "0" and "1" in the abbreviations indicate the least and most significant portions of the registers or descriptors. Table 3-1 lists the naming convention abbreviations for descriptors.

##### 3.2.3 Buffer Management Base Addresses

The SONIC-16 uses three areas in memory to store descriptor information: the Transmit Descriptor Area (TDA), Receive Descriptor Area (RDA), and the Receive Resource Area (RRA). The SONIC-16 accesses these areas by concatenating a 16-bit base address register with a 16-bit offset register. The base address register supplies a fixed upper 8 bits of address and the offset registers provide the lower 16 bits of address. The base address registers are the Upper Transmit Descriptor Address (UTDA), Upper Receive Descriptor Address (URDA), and the Upper Receive Resource Address (URRA) registers. The corresponding offset registers are shown below.

Upper Address Registers	Offset Registers
URRA	RSA, REA, RWP, RRP
URDA	CRDA
UTDA	CTDA

See Table 3-1 for definition of register mnemonics.

*Figure 3-1* shows an example of the Transmit Descriptor Area and the Receive Descriptor Area being located by the UTDA and URDA registers. The descriptor areas, RDA, TDA, and RRA are allowed to have the same base address. i.e., URRA=URDA=UTDA. Care, however, must be taken to prevent these areas from overwriting each other.



### 3.0 Buffer Management (Continued)

TABLE 3-1. Descriptor Abbreviations

TRANSMIT AND RECEIVE AREAS	
RRA	Receive Resource Area
RDA	Receive Descriptor Area
RBA	Receive Buffer Area
TDA	Transmit Descriptor Area
TBA	Transmit Buffer Area
BUFFER MANAGEMENT REGISTERS	
RSA	Resource Start Area Register
REA	Resource End Area Register
RRP	Resource Read Pointer Register
RWP	Resource Write Pointer Register
CRDA	Current Receive Descriptor Address Register
CRBA0,1	Current Receive Buffer Address Register
TCBA0,1	Temporary Current Buffer Address Register
RBWC0,1	Remaining Buffer Word Count Register
TRBWC0,1	Temporary Remaining Buffer Word Count Register
EOBC	End of Buffer Count Register
TPS	Transmit Packet Size Register
TSA0,1	Transmit Start Address Register
CTDA	Current Transmit Descriptor Address Register

BUFFER MANAGEMENT REGISTERS (Continued)	
TFC	Transmit Fragment Count Register
TFS	Transmit Fragment Size Register
UTDA	Upper Transmit Descriptor Address Register
URRA	Upper Receive Resource Address Register
URDA	Upper Receive Descriptor Address Register
TRANSMIT AND RECEIVE DESCRIPTORS	
RXsrc.buff_ptr0,1	Buffer Pointer Field in the RRA
RXsrc.buff_wc0,1	Buffer Word Count Fields in the RRA
RXpkt.status	Receive Status Field in the RDA
RXpkt.byte_count	Packet Byte Count Field in the RDA
RXpkt.buff_ptr0,1	Buffer Pointer Fields in the RDA
RXpkt.link	Receive Descriptor Link Field in RDA
RXpkt.in_use	"In Use" Field in RDA
TXpkt.frag_count	Fragment Count Field in TDA
TXpkt.pkt_size	Packet Size Field in TDA
TXpkt.pkt_ptr0,1	Packet Pointer Fields in TDA
TXpkt.frag_size	Fragment Size Field in TDA
TXpkt.link	Transmit Descriptor Link Field in TDA

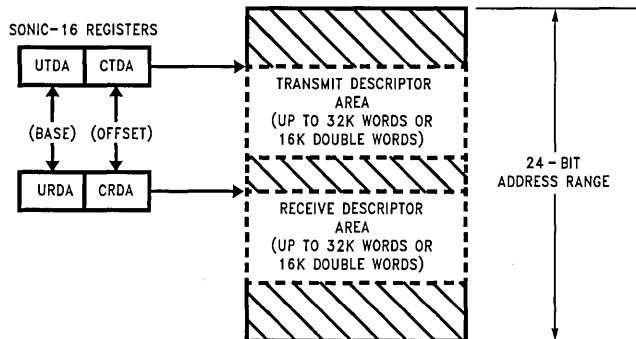


FIGURE 3-1. Transmit and Receive Descriptor Area Pointers

TL/F/11722-10

### 3.0 Buffer Management (Continued)

#### 3.3 DESCRIPTOR DATA ALIGNMENT

All fields used by descriptors (RXpkt.xxx, RXrsrc.xxx, and TXpkt.xxx) are word quantities (16-bit) and must be aligned to word boundaries (A0=0). The Receive Buffer Area (RBA) must also be aligned to a word boundary. The fragments in the Transmit Buffer Area (TBA), however, may be aligned on any arbitrary byte boundary.

All descriptor areas follow little endian byte ordering, even when BMODE = 1.

#### 3.4 RECEIVE BUFFER MANAGEMENT

The Receive Buffer Management operates on three areas in memory into which data, status, and control information are written during reception (Figure 3-2). These three areas must be initialized (section 3.4.4) before enabling the receiver (setting the RXEN bit in the Command register). The receive resource area (RRA) contains descriptors that locate receive buffer areas in system memory. These descriptors are denoted by R1, R2, etc. in Figure 3-2. Packets (denoted by P1, P2, etc.) can then be buffered into the corresponding RBAs. Depending on the size of each buffer area and the size of the packet(s), multiple or single packets are buffered into each RBA. The receive descriptor area (RDA) contains status and control information for each packet (D1, D2, etc. in Figure 3-2) corresponding to each received packet (D1 goes with P1, D2 with P2, etc.).

When a packet arrives, the address recognition logic checks the address for a Physical, Multicast, or Broadcast match and if the packet is accepted, the SONIC-16 buffers the packet contiguously into the selected Receive Buffer Area (RBA). Because of the previous end-of-packet processing, the SONIC-16 assures that the complete packet is written into a single contiguous block. When the packet ends, the SONIC-16 writes the receive status, byte count, and location of the packet into the Receive Descriptor Area (RDA). The SONIC-16 then updates its pointers to locate the next available descriptor and checks the remaining words available in the RBA. If sufficient space remains, the SONIC-16 buffers the next packet immediately after the previous pack-

et. If the current buffer is out of space the SONIC-16 fetches a Resource descriptor from the Receive Resource Area (RRA) acquiring an additional buffer that has been previously allocated by the system.

#### 3.4.1 Receive Resource Area (RRA)

As buffer memory is consumed by the SONIC-16 for storing data, the Receive Resource Area (RRA) provides a mechanism that allows the system to allocate additional buffer space for the SONIC-16. The system loads this area with resource descriptors that the SONIC-16, in turn, reads as its current buffer space is used up. Each resource descriptor consists of a 23-bit buffer pointer locating the starting point of the RBA and a 32-bit Word Count that indicates the size of the buffer in words (2 bytes per word). The buffer pointer and word count are contiguously located using the format shown in Figure 3-3 with each component composed of 16-bit fields. The SONIC-16 stores this information internally and concatenates the corresponding fields to create 23- and 32-bit long words for the buffer pointer and word count.

The SONIC-16 organizes the RRA as a circular queue for efficient processing of descriptors. Four registers define the RRA. The first two, the Resource Start Area (RSA) and the Resource End Area (REA) registers, determine the starting and ending locations of the RRA, and the other two registers update the RRA. The system adds descriptors at the address specified by the Resource Write Pointer (RWP), and the SONIC-16 reads the next descriptor designated by the Resource Read Pointer (RRP). The RRP is advanced 4 words after the SONIC-16 finishes reading the RRA and automatically wraps around to the beginning of the RRA once the end has been reached. When a descriptor in the RRA is read, the RXrsrc.buf\_pt0,1 is loaded into the CRBA0,1 registers and the RXrsrc.buf\_wc0,1 is loaded into the RBWC0,1 registers.

The alignment of the RRA is confined to word boundaries (A0 is always zero).

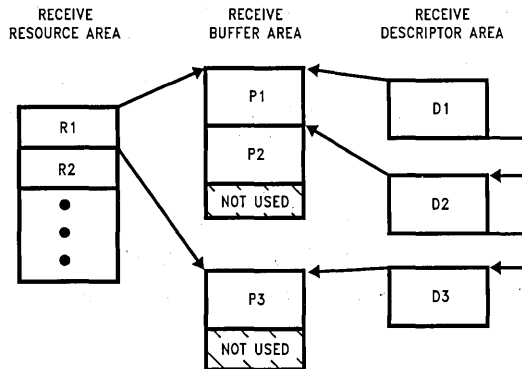


FIGURE 3-2. Overview of Receive Buffer Management

TL/F/11722-11

### 3.0 Buffer Management (Continued)

#### 3.4.2 Receive Buffer Area (RBA)

The SONIC-16 stores the actual data of a received packet in the RBA. The RBAs are designated by the resource descriptors in the RRA as described above. The `RXsrc.buf_wc0,1` fields of the RRA indicate the length of the RBA. When the SONIC-16 gets a RBA from the RRA, the `RXsrc.buf_wc0,1` values are loaded into the Remaining Buffer Word Count registers (`RBWC0,1`). These registers keep track of how much space (in words) is left in the buffer. When a packet is buffered in a RBA, it is buffered contiguously (the SONIC-16 will not scatter a packet into multiple buffers or fragments). Therefore, if there is not enough space left in a RBA after buffering a packet to buffer at least one more maximum sized packet (the maximum legal sized packet expected to be received from the network), a new buffer must be acquired. The End of Buffer Count (EOBC) register is used to tell the SONIC-16 the maximum packet size that the SONIC-16 will need to buffer.

##### 3.4.2.1 End of Buffer Count (EOBC)

The EOBC is a boundary in the RBA based from the bottom of the buffer. The value written into the EOBC is the maximum expected size (in words) of the network packet that the SONIC-16 will have to buffer. This word count creates a line in the RBA that, when crossed, causes the SONIC-16 to fetch a new RBA resource from the RRA.

**Note:** The EOBC is a word count, not a byte count.

##### 3.4.2.2 Buffering the Last Packet in an RBA

At the start of reception, the SONIC-16 stores the packet beginning at the Current Receive Buffer Address (`CRBA0,1`) and continues until the reception is complete. Concurrent with reception, the SONIC-16 decrements the Remaining Buffer Word Count (`RBWC0,1`) by one. At the end of reception, if the packet has crossed the EOBC boundary, the SONIC-16 knows that the next packet might not fit in the RBA. This check is done by comparing the `RBWC0,1` registers with the EOBC. If `RBWC0,1` is less than the EOBC (the last packet buffered has crossed the EOBC boundary), the SONIC-16 fetches the next resource descriptor in the RRA. If `RBWC0,1` is greater than or equal to the EOBC (the EOBC boundary has not been crossed) the next packet reception continues at the present location pointed to by `CRBA0,1` in the same RBA. *Figure 3-4* illustrates the SONIC-16's actions for (1)  $RBWC0,1 \geq EOBC$  and (2)  $RBWC0,1 < EOBC$ . See Section 3.4.4.4 for specific information about setting the EOBC.

**Note:** It is important that the EOBC boundary be "crossed." In other words, case #1 in *Figure 3-4* must exist before case #2 exists. If case #2 occurs without case #1 having occurred first, the test for  $RBWC0,1 < EOBC$  will not work properly and the SONIC-16 will not fetch a new buffer. The result of this will be a buffer overflow (RBAE in the Interrupt Status Register, section 4.3.6).

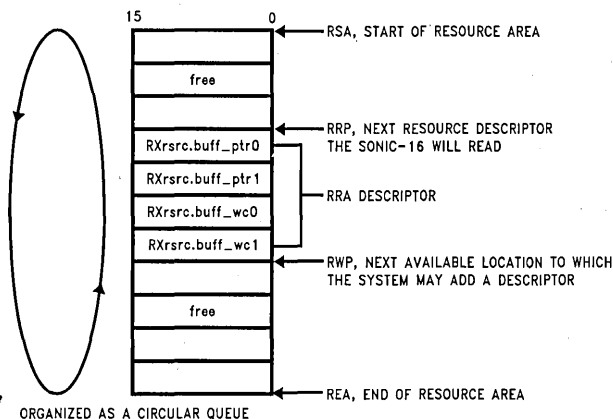
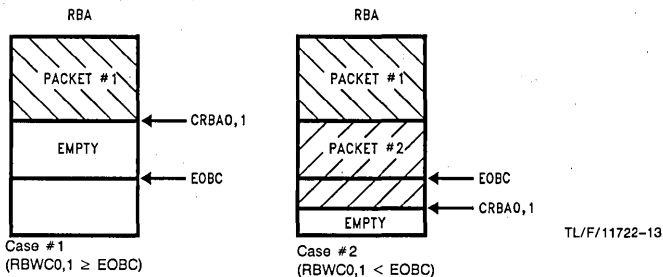


FIGURE 3-3. Receive Resource Area Format



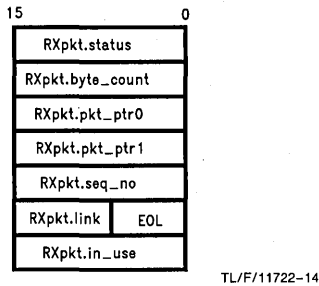
Case #1: SONIC-16 buffers next packet in same RBA.  
 Case #2: SONIC-16 detects an exhausted RBA and will buffer the next packet in another RBA.

FIGURE 3-4. Receive Buffer Area

### 3.0 Buffer Management (Continued)

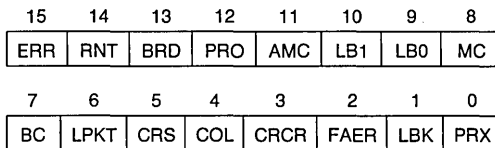
#### 3.4.3 Receive Descriptor Area (RDA)

After the SONIC-16 buffers a packet to memory, it writes 5 words of status and control information into the RDA, reads the link field to the next receive descriptor and writes to the in use field of the current descriptor. Each receive descriptor consists of the following sections (*Figure 3-5*).



**FIGURE 3-5. Receive Descriptor Format**

**receive status:** indicates status of the received packet. The SONIC-16 writes the Receive Control register into this field. *Figure 3-6* shows the receive status format. This field is loaded from the contents of the Receive Control register. Note that ERR, RNT, BRD, PRO, and AMC are configuration bits and are programmed during initialization. See Section 4.3.3 for the description of the Receive Control register.

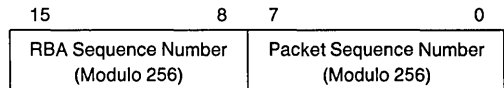


**FIGURE 3-6. Receive Status Format**

**byte count:** gives the length of the complete packet from the start of Destination Address to the end of FCS.

**packet pointer:** a 23-bit pointer that locates the packet in the RBA. The SONIC-16 writes the contents of the CRBA0,1 registers into this field.

**sequence numbers:** this field displays the contents of two 8-bit counters (modulo 256) that sequence the RBAs used and the packets buffered. These counters assist the system in determining when an RBA has been completely processed. The sequence numbers allow the system to tally the packets that have been processed within a particular RBA. There are two sequence numbers that describe a packet: the RBA Sequence Number and the Packet Sequence Number. When a packet is buffered to memory, the SONIC-16 maintains a single RBA Sequence Number for all packets in an RBA and sequences the Packet Number for succeeding packets in the RBA. When the SONIC-16 uses the next RBA, it increments the RBA Sequence Number and clears the Packet Sequence Number. The RBA's sequence counter is not incremented when the read RRA command is issued in the Command register. The format of the Receive Sequence Numbers are shown in *Figure 3-7*. These counters are reset during hardware reset or by writing zero to them.



**FIGURE 3-7. Receive Sequence Number Format**

**receive link field:** a 15-bit pointer (A15-A1) that locates the next receive descriptor. The LSB of this field is the End Of List (EOL) bit, and indicates the last descriptor in the list. (Initialized by the system.)

**in use field:** this field provides a handshake between the system and the SONIC-16 to indicate the ownership of the descriptor. When the system avails a descriptor to the SONIC-16, it writes a non-zero value into this field. The SONIC-16, in turn, sets this field to all "0's" when it has finished processing the descriptor. (That is, when the CRDA register has advanced to the next receive descriptor.) Generally, the SONIC-16 releases control after writing the status and control information into the RDA. If, however, the SONIC-16 has reached the last descriptor in the list, it maintains ownership of the descriptor until the system has appended additional descriptors to the list. The SONIC-16 then relinquishes control after receiving the next packet. (See Section 3.4.6.1 for details on when the SONIC-16 writes to this field.) The receive packet descriptor format is shown in *Figure 3-5*.

#### 3.4.4 Receive Buffer Management Initialization

The Receive Resource, Descriptor, and Buffer areas (RRA, RDA, RBA) in memory and the appropriate SONIC-16 registers must be properly initialized before the SONIC-16 begins buffering packets. This section describes the initialization process.

##### 3.4.4.1 Initializing The Descriptor Page

All descriptor areas (RRA, RDA, and TDA) used by the SONIC-16 reside within areas up to 32k (word) pages. This page may be placed anywhere within the 23-bit address range by loading the upper 8 address lines into the UTDA, URDA, and URRR registers.

##### 3.4.4.2 Initializing The RRA

The initialization of the RRA consists of loading the four SONIC-16 RRA registers and writing the resource descriptor information to memory.

The RRA registers are loaded with the following values.

**Resource Start Area (RSA) register:** The RSA is loaded with the lower 16-bit address of the beginning of the RRA.

**Resource End Area (REA) register:** The REA is loaded with the lower 16-bit address of the end of the RRA. The end of the RRA is defined as the address of the last RXsrc.ptr0 field in the RRA plus 4 words (*Figure 3-3*).

**Resource Read Pointer (RRP) register:** The RRP is loaded with the lower 16-bit address of the first resource descriptor the SONIC-16 reads.

**Resource Write Pointer (RWP) register:** The RWP is loaded with the lower 16-bit address of the next vacant location where a resource descriptor will be placed by the system.

**Note:** The RWP register must only point to either (1) the RXsrc.ptr0 field of one of the RRA Descriptors, (2) the memory address that the RSA points to (the start of the RRA), or (3) the memory address that the REA points to (the end of the RRA). When the RWP = RRP comparison is made, it is performed after the complete RRA descriptor has been read and not during the fetch. Failure to set the RWP to any of the above values prevents the RWP = RRP comparison from ever becoming true.

### 3.0 Buffer Management (Continued)

All RRA registers are concatenated with the URRRA register for generating the full 23-bit address.

The resource descriptors that the system writes to the RRA consists of four fields: (1) RXsrc.buf\_ptr0, (2) RXsrc.buf\_ptr1, (3) RXsrc.buf\_wc0, and (4) RXsrc.buf\_wc1. The fields must be contiguous (they cannot straddle the end points) and are written in the order shown in Figure 3-8. The "0" and "1" in the descriptors denote the least and most significant portions for the Buffer Pointer and Word Count. The first two fields supply the 23-bit starting location of the Receive Buffer Area (RBA), and the second two define the number of 16-bit words that the RBA occupies. Note that a restriction applies to the Buffer Pointer and Word Count. The Buffer Pointer must be pointing to a word boundary. Note also that the descriptors must be properly aligned in the RRA as discussed in Section 3.3.

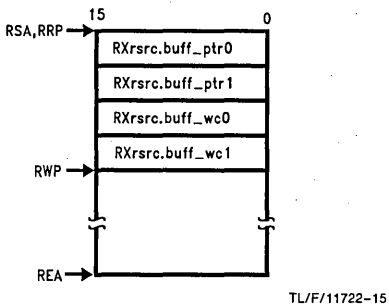


FIGURE 3-8. RRA Initialization

After configuring the RRA, the RRA Read command (setting RRRRA bit in the Command register) may be given. This command causes the SONIC-16 to read the RRA descriptor in a single block operation, and load the following registers (see Section 4.2 for register mnemonics):

- CRBA0 register ← RXsrc.buf\_ptr0
- CRBA1 register ← RXsrc.buf\_ptr1
- RBWC0 register ← RXsrc.buf\_wc0
- RBWC1 register ← RXsrc.buf\_wc1

When the command has completed, the RRRRA bit in the Command register is reset to "0". Generally this command is only issued during initialization. At all other times, the RRA is automatically read as the SONIC-16 finishes using an RBA.

#### 3.4.4.3 Initializing The RDA

To accept multiple packets from the network, the receive packet descriptors must be linked together via the RXpkt.link fields. Each link field must be written with a 15-bit (A15-A1) pointer to locate the beginning of the next descriptor in the list. The LSB of the RXpkt.link field is the End of List (EOL) bit and is used to indicate the end of the descriptor list. EOL = 1 for the last descriptor and EOL = 0 for the first or middle descriptors. The RXpkt.in\_use field indicates whether the descriptor is owned by the SONIC-16. The system writes a non-zero value to this field when the descriptor is available, and the SONIC-16 writes all "0's"

when it finishes using the descriptor. At startup, the Current Receive Descriptor Address (CRDA) register must be loaded with the address of the first RXpkt.status field in order for the SONIC-16 to begin receive processing at the first descriptor. An example of two descriptors linked together is shown in Figure 3-9. The fields initialized by the system are displayed in larger type. The other fields are written by the SONIC-16 after a packet is accepted. The RXpkt.in\_use field is first written by the system, and then by the SONIC-16. Note that the descriptors must be aligned properly as discussed in section 3.3. Also note that the URDA register is concatenated with the CRDA register to generate the full 23-bit address.

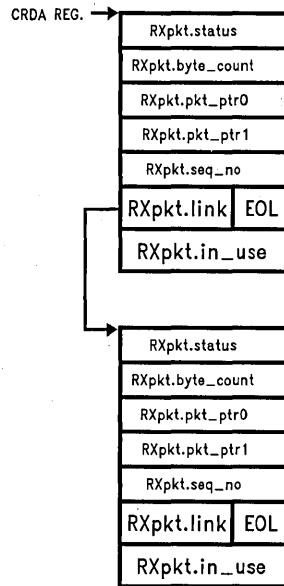


FIGURE 3-9. RDA Initialization Example

#### 3.4.4.4 Initializing the Lower Boundary of the RBA

A "false bottom" is set in the RBA by loading the End Of Buffer Count (EOBC) register with a value equal to the maximum size packet in words (16 bits) that may be received. This creates a lower boundary in the RBA. Whenever the Remaining Buffer Word Count (RBWC0,1) registers decrement below the EOBC register, the SONIC-16 buffers the next packet into another RBA. This also guarantees that a packet is always contiguously buffered into a single Receive Buffer Area (RBA). The SONIC-16 does not buffer a packet into multiple RBAs.

After a hardware reset, the EOBC register is automatically initialized to 2F8h (760 words or 1520 bytes).

Sometimes it may be desired to buffer a single packet per RBA. When doing this, it is important to set EOBC and the buffer size correctly. The suggested practice is to set EOBC to a value that is at least 2 bytes less than the buffer size.

### 3.0 Buffer Management (Continued)

An example would be EOBC = 759 words (1518 bytes) and the buffer size set to 760 words (1520 bytes). The buffer can be any size, but as long as the EOBC is 1 word less than the buffer size, only one packet will be buffered in that RBA.

**Note 1:** It is possible to filter out most oversized packets by setting the buffer size to 759 words (1518 bytes). EOBC would be set to 758 words (1516 bytes) for both cases. With this configuration, any packet over 1518 bytes, will not be completely buffered because the packet will overflow the buffer. When a packet overflow occurs, a Receive Buffer Area Exceeded interrupt (RBAE in the Interrupt Status Register, Section 4.3.6) will occur.

**Note 2:** When buffering one packet per buffer, it is suggested that the values in Note 1 above be used. Since the minimum legal sized Ethernet packet is 64 bytes, however, it is possible to set EOBC as much as 64 bytes less than the buffer size and still end up with one packet per buffer. Figure 3-10 shows this "range."

#### 3.4.5 Beginning of Reception

At the beginning of reception, the SONIC-16 checks its internally stored EOL bit from the previous RXpkt.link field for a "1". If the SONIC-16 finds EOL = 1, it recognizes that after the previous reception, there were no more remaining receive packet descriptors. It re-reads the same RXpkt.link field to check if the system has updated this field since the last reception. If the SONIC-16 still finds EOL = 1, reception ceases. (See Section 3.5 for adding descriptors to the list.) Otherwise, the SONIC-16 begins storing the packet in the RBA starting at the Current Receive Buffer Address (CRBA0,1) registers and continues until the packet has completed. Concurrent with the packet reception, the Remaining Buffer Word Count (RBWC0,1) registers are decremented after each word is written to memory. This register determines the remaining words in the RBA at the end of reception.

#### 3.4.6 End of Packet Processing

At the end of a reception, the SONIC-16 enters its end of packet processing sequence to determine whether to accept or reject the packet based on receive errors and packet size. At the end of reception the SONIC-16 enters one of the following two sequences:

- Successful reception sequence
- Buffer recovery for runt packets or packets with errors

#### 3.4.6.1 Successful Reception

If the SONIC-16 accepts the packet, it first writes 5 words of descriptor information in the RDA beginning at the address pointed to by the Current Receive Descriptor Address (CRDA) register. It then reads the RXpkt.link field to advance the CRDA register to the next receive descriptor. The SONIC-16 also checks the EOL bit for a "1" in this field. If EOL = 1, no more descriptors are available for the SONIC-16. The SONIC-16 recovers the address of the current RXpkt.link field (from a temporary register) and generates a "Receive Descriptors Exhausted" indication in the Interrupt Status register. (See Section 3.4.7 on how to add descriptors.) The SONIC-16 maintains ownership of the descriptor by *not* writing to the RXpkt.in\_use field. Otherwise, if EOL = 0, the SONIC-16 advances the CRDA register to the next descriptor and resets the RXpkt.in\_use field to all "0's".

The SONIC-16 accesses the complete 7 word RDA descriptor in a single block operation.

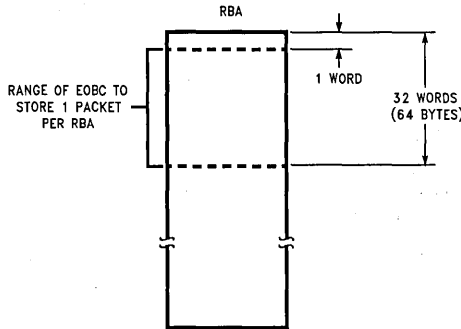
The SONIC-16 also checks if there is remaining space in the RBA. The SONIC-16 compares the Remaining Buffer Word Count (RBWC0,1) registers with the static End Of Buffer Count (EOBC). If the RBWC is less than the EOBC, a maximum sized packet will no longer fit in the remaining space in the RBA; hence, the SONIC-16 fetches a resource descriptor from the RRA and loads its registers with the pointer and word count of the next available RBA.

#### 3.4.6.2 Buffer Recovery for Runt Packets or Packets with Errors

If a runt packet (less than 64 bytes) or packet with errors arrives and the Receive Control register has been configured to not accept these packets, the SONIC-16 recovers its pointers back to the original positions. The CRBA0,1 registers are not advanced and the RBWC0,1 registers are not decremented. The SONIC-16 recovers its pointers by maintaining a copy of the buffer address in the Temporary Receive Buffer Address registers (TRBA0,1). The SONIC-16 recovers the value in the RBWC0,1 registers from the Temporary Buffer Word Count registers (TBWC0,1).

#### 3.4.7 Overflow Conditions

When an overflow condition occurs, the SONIC-16 halts its DMA operations to prevent writing into unauthorized memory. The SONIC-16 uses the Interrupt Status register (ISR) to indicate three possible overflow conditions that can occur



$$\text{Range of EOBC} = (\text{RXsrc.wc0,1} - 2 \text{ to } \text{RXsrc.wc0,1} - 32)$$

TL/F/11722-17

FIGURE 3-10. Setting EOBC for Single Packet RBA

### 3.0 Buffer Management (Continued)

when its receive resources have been exhausted. The system should respond by replenishing the resources that have been exhausted. These overflow conditions (Descriptor Resources Exhausted, Buffer Resources Exhausted, and RBA Limit Exceeded) are indicated in the Interrupt Status register and are detailed as follows:

**Descriptor Resources Exhausted:** This occurs when the SONIC-16 has reached the last receive descriptor in the list, meaning that the SONIC-16 has detected EOL = 1. The system must supply additional descriptors for continued reception. The system can do this in one of two ways: 1) appending descriptors to the existing list, or 2) creating a separate list.

1) Appending descriptors to the existing list. This is the easiest and preferred way. To do this, the system, after creating the new list, joins the new list to the existing list by simply writing the beginning address of the new list into the RXpkt.link field and setting EOL = 0. At the next reception, the SONIC-16 re-reads the last RXpkt.link field, and updates its CRDA register to point to the next descriptor.

2) Creating a separate list. This requires an additional step because the lists are not joined together and requires that the CRDA register be loaded with the address of the RXpkt.link field in the new list.

During this overflow condition, the SONIC-16 maintains ownership of the descriptor (RXpkt.in\_use  $\neq$  00h) and waits for the system to add additional descriptors to the list. When the system appends more descriptors, the SONIC-16 releases ownership of the descriptor after writing 0000h to the RXpkt.in\_use field.

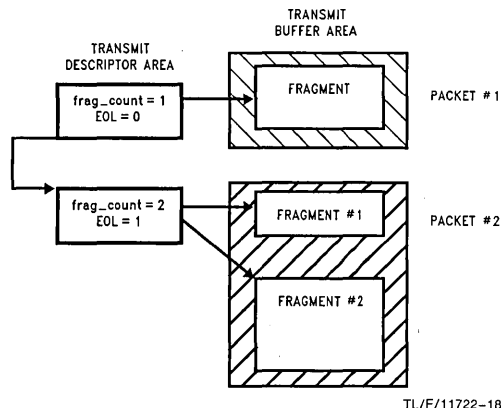
**Buffer Resources Exhausted:** This occurs when the SONIC-16 has detected that the Resource Read Pointer (RRP) and Resource Write Pointer (RWP) registers are equal (i.e., all RRA descriptors have been exhausted). The RBE bit in the Interrupt Status register is set when the SONIC-16 finishes using the second to last receive buffer and reads the last RRA descriptor. Actually, the SONIC-16 is not truly out of resources, but gives the system an early warning of an impending out of resources condition. To continue reception after the last RBA is used, the system must supply additional RRA descriptor(s), update the RWP register, and clear the RBE bit in the ISR. The SONIC-16 rereads the RRA after this bit is cleared.

**RBA Limit Exceeded:** This occurs when a packet does not completely fit within the remaining space of the RBA. This can occur if the EOBC register is not programmed to a value greater than the largest packet that can be received. When this situation occurs, the packet is truncated and the SONIC-16 reads the RRA to obtain another RBA. Indication of an RBA limit being exceeded is signified by the Receive Buffer Area Exceeded (RBAE) interrupt being set (see section 4.3.6). An RDA will not be set up for the truncated packet and the buffer space will not be re-used. To rectify this potential overflow condition, the EOBC register must be loaded with a value equal to or greater than the largest packet that can be accepted. See Section 3.4.2.

#### 3.5 TRANSMIT BUFFER MANAGEMENT

To begin transmission, the system software issues the Transmit command (TXP = 1 in the CR). The Transmit Buffer Management uses two areas in memory for transmitting packets (Figure 3-11), the Transmit Descriptor Area (TDA)

and the Transmit Buffer Area (TBA). During transmission, the SONIC-16 fetches control information from the TDA, loads its appropriate registers, and then transmits the data from the TBA. When the transmission is complete, the SONIC-16 writes the status information in the TDA. From a single transmit command, packets can either be transmitted singly or in groups if several descriptors have been linked together.



TL/F/11722-18

FIGURE 3-11. Overview of Transmit Buffer Management

#### 3.5.1 Transmit Descriptor Area (TDA)

The TDA contains descriptors that the system has generated to exchange status and control information. Each descriptor corresponds to a single packet and consists of the following 16-bit fields.

**TXpkt.status:** This field is written by the SONIC-16 and provides status of the transmitted packet. See Section 3.5.1.2 for more details.

**TXpkt.config:** This field allows programming the SONIC-16 to one of the various transmit modes. The SONIC-16 reads this field and loads the corresponding configuration bits (PINTR, POWC, CRCL, and EXDIS) into the Transmit Control register. See Section 3.5.1.1 for more details.

**TXpkt.pkt\_size:** This field contains the byte count of the entire packet

**TXpkt.frag\_count:** This field contains the number of fragments the packet is segmented into.

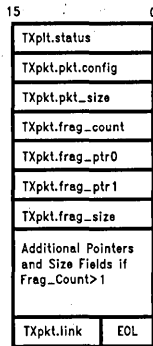
**TXpkt.frag\_ptr0,1:** This field contains a 23-bit pointer which locates the packet fragment to be transmitted in the Transmit Buffer Area (TBA). This pointer is not restricted to any byte alignment.

**TXpkt.frag\_size:** This field contains the byte count of the packet fragment. The minimum fragment size is 1 byte.

**TXpkt.link:** This field contains a 15-bit pointer (A15-A1) to the next TDA descriptor. The LSB, the End Of List (EOL) bit, indicates the last descriptor in the list when set to a "1". When descriptors have been linked together, the SONIC-16 transmits back-to-back packets from a single transmit command.

The data of the packet does not need to be contiguous, but can exist in several locations (fragments) in memory. In this case, the TXpkt.frag\_count field is greater than one, and additional TXpkt.frag\_ptr0,1 and TXpkt.frag\_size fields corresponding to each fragment are used. The descriptor format is shown in Figure 3-12.

### 3.0 Buffer Management (Continued)

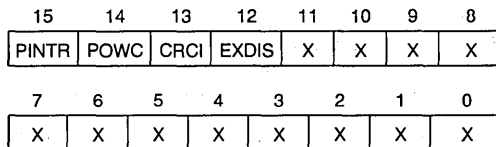


TL/F/11722-19

**FIGURE 3-12. Transmit Descriptor Area**

#### 3.5.1.1 Transmit Configuration

The TXpkt.config field allows the SONIC-16 to be programmed into one of the transmit modes before each transmission. At the beginning of each transmission, the SONIC-16 reads this field and loads the PINTR, POWC, CRCI, and EXDIS bits into the Transmit Control register (TCR). The configuration bits in the TCR correspond directly with the bits in the TXpkt.config field as shown in *Figure 3-13*. See Section 4.3.4 for the description on the TCR.

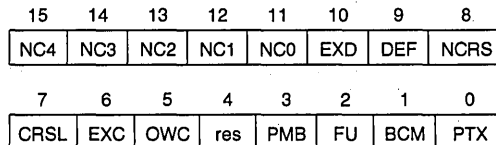


Note: x = don't care

**FIGURE 3-13. TXpkt.config Field**

#### 3.5.1.2 Transmit Status

At the end of each transmission the SONIC-16 writes the status bits (<10:0>) of the Transmit Control Register (TCR) and the number of collisions experienced during the transmission into the TXpkt.status field (*Figure 3-14*, res = reserved). Bits NC4-NC0 indicate the number of collisions where NC4 is the MSB. See Section 4.3.4 for the description of the TCR.



**FIGURE 3-14. TXpkt.status Field**

#### 3.5.2 Transmit Buffer Area (TBA)

The TBA contains the fragments of packets that are defined by the descriptors in the TDA. A packet can consist of a single fragment or several fragments, depending upon the fragment count in the TDA descriptor. The fragments also can reside anywhere within the full 23-bit address range, and be aligned to any byte boundary. When an odd byte boundary is given, the SONIC-16 automatically begins reading data at the corresponding word boundary. The SONIC-16 ignores the extraneous bytes which are written into the

FIFO during odd byte alignment fragments. The minimum allowed fragment size is 1 byte. *Figure 3-11* shows the relationship between the TDA and the TBA for single and multi-fragmented packets.

#### 3.5.3 Preparing To Transmit

All fields in the TDA descriptor and the Current Transmit Descriptor Address (CTDA) register of the SONIC-16 must be initialized before the Transmit Command (setting the TXP bit in the Command register) can be issued. If more than one packet is queued, the descriptors must be linked together with the TXpkt.link field. The last descriptor must have EOL = 1 and all other descriptors must have EOL = 0. To begin transmission, the system loads the address of the first TXpkt.status field into the CTDA register. Note that the upper 8-bits of address are loaded in the Upper Transmit Descriptor (UTDA) register. The user performs the following transmit initialization.

- 1) Initialize the TDA
- 2) Load the CTDA register with the address of the first transmit descriptor
- 3) Issue the transmit command

Note that if the Source Address of the packet being transmitted is not in the CAM, the Packet Monitored Bad (PMB) bit in the TXpkt.status field will be set (see Section 4.3.4).

##### 3.5.3.1 Transmit Process

When the Transmit Command (TXP = 1 in the Command register) is issued, the SONIC-16 fetches the control information in the TDA descriptor, loads its appropriate registers (shown below) and begins transmission. (See Section 4.2 for register mnemonics.)

- TCR ← TXpkt.config
- TPS ← TXpkt.pkt\_size
- TFC ← TXpkt.frag\_count
- TSA0 ← TXpkt.frag\_ptr0
- TSA1 ← TXpkt.frag\_ptr1
- TFS ← TXpkt.frag\_size
- CTDA ← TXpkt.link

(CTDA is loaded after all fragments have been read and successfully transmitted. If the halt transmit command is issued (HTX bit in the Command register is set) the CTDA register is not loaded.)

During transmission, the SONIC-16 reads the packet descriptor in the TDA and transmits the data from the TBA. If TXpkt.frag\_count is greater than one, the SONIC-16, after finishing transmission of the fragment, fetches the next TXpkt.frag\_ptr0,1 and TXpkt.frag\_size fields and transmits the next fragment. This process continues until all fragments of a packet are transmitted. At the end of packet transmission, status is written in to the TXpkt.status field. The SONIC-16 then reads the TXpkt.link field and checks if EOL = 0. If it is "0", the SONIC-16 fetches the next descriptor and transmits the next packet. If EOL = 1 the SONIC-16 generates a "Transmission Done" indication in the Interrupt Status register and resets the TXP bit in the Command register.

In the event of a collision, the SONIC-16 recovers its pointer in the TDA and retransmits the packet up to 15 times. The SONIC-16 maintains a copy of the CTDA register in the Temporary Transmit Descriptor Address (TTDA) register.

The SONIC-16 performs a block operation of 6, 3, or 2 accesses in the TDA, depending on where the SONIC-16 is in the transmit process. For the first fragment, it reads the



### 3.0 Buffer Management (Continued)

TXpkt.config to TXpkt.frag\_size (6 accesses). For the next fragment, if any, it reads the next 3 fields from TXpkt.frag\_ptr0 to TXpkt.frag\_size (3 accesses). At the end of transmission it writes the status information to TXpkt.status and reads the TXpkt.link field (2 accesses).

#### 3.5.3.2 Transmit Completion

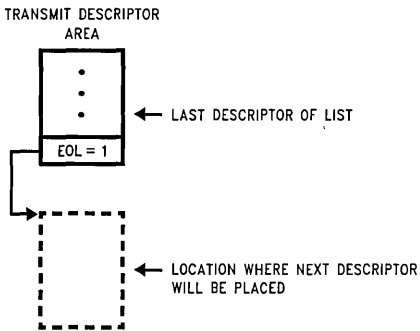
The SONIC-16 stops transmitting under two conditions. In the normal case, the SONIC-16 transmits the complete list of descriptors in the TDA and stops after it detects EOL = 1. In the second case, certain transmit errors cause the SONIC-16 to abort transmission. If *FIFO Underrun*, *Byte Count Mismatch*, *Excessive Collision*, or *Excessive Deferral* (if enabled) errors occur, transmission ceases. The CTDA register points to the last packet transmitted. The system can also halt transmission under software control by setting the HTX bit in the Command register. Transmission halts after the SONIC-16 writes to the TXpkt.status field.

#### 3.5.4 Dynamically Adding TDA Descriptors

Descriptors can be dynamically added during transmission without halting the SONIC-16. The SONIC-16 can also be guaranteed to transmit the complete list including newly appended descriptors (barring any transmit abort conditions) by observing the following rule: The last TXpkt.link field must point to the next location where a descriptor will be added (see step 3 below and *Figure 3-15*). The procedure for appending descriptors consists of:

1. Creating a new descriptor with its TXpkt.link pointing to the next vacant descriptor location and its EOL bit set to a "1".
2. Resetting the EOL bit to a "0" of the previously last descriptor.
3. Re-issuing the Transmit command (setting the TXP bit in the Command register).

Step 3 assures that the SONIC-16 will transmit all the packets in the list. If the SONIC-16 is currently transmitting, the Transmit command has no effect and continues transmitting until it detects EOL = 1. If the SONIC-16 had just finished transmitting, it continues transmitting from where it had previously stopped.



TL/F/11722-20

FIGURE 3-15. Initializing Last Link Field

### 4.0 SONIC-16 Registers

The SONIC-16 contains two sets of registers: The status/control registers and the CAM memory cells. The status/control registers are used to configure, control, and monitor SONIC-16 operation. They are directly addressable registers and occupy 64 consecutive address locations in the system memory space (selected by the RA5-RA0 address pins). There are a total of 64 status/control registers divided into the following categories:

**User Registers:** These registers are accessed by the user to configure, control, and monitor SONIC-16 operation. These are the only SONIC-16 registers the user needs to access. *Figure 4-3* shows the programmer's model and *Table 4-1* lists the attributes of each register.

**Internal Use Registers:** These registers (*Table 4-2*) are used by the SONIC-16 during normal operation and are not intended to be accessed by the user.

**National Factory Test Registers:** These registers (*Table 4-3*) are for National factory use only and should never be accessed by the user. Accessing these registers during normal operation can cause improper functioning of the SONIC-16.

#### 4.1 THE CAM UNIT

The CAM unit memory cells are indirectly accessed by programming the CAM descriptor area in system memory and issuing the LCAM command (setting the LCAM bit in the Control register). The CAM cells do not occupy address locations in register space and, thus, are not accessible through the RA5-RA0 address pins. The CAM control registers, however, are part of the user register set and must be initialized before issuing the LCAM command (see *Section 4.3.10*).

The Content Addressable Memory (CAM) consists of sixteen 48-bit entries for complete address filtering (*Figure 4-1*) of network packets. Each entry corresponds to a 48-bit destination address that is user programmable and can contain any combination of Multicast or Physical addresses. Each entry is partitioned into three 16-bit CAM cells accessible through CAM Address Ports (CAP 2, CAP 1 and CAP 0) with CAP0 corresponding to the least significant 16 bits of the Destination Address and CAP2 corresponding to the most significant bits. The CAM is accessed in a two step process. First, the CAM Entry Pointer is loaded to point to one of the 16 entries. Then, each of the CAM Address Ports is accessed to select the CAM cell. The 16 user programmable CAM entries can be masked out with the CAM Enable register (see *section 4.3.10*).

**Note:** It is not necessary to program a broadcast address into the CAM when it is desired to accept broadcast packets. Instead, to accept broadcast packets, set the BRD bit in the Receive Control register. If the BRD bit has been set, the CAM is still active. This means that it is possible to accept broadcast packets at the same time as accepting packets that match physical addresses in the CAM.

#### 4.1.1 The Load CAM Command

Because the SONIC-16 uses the CAM for a relatively long period of time during reception, it can only be written to via the CAM Descriptor Area (CDA) and is only readable when

## 4.0 SONIC-16 Registers (Continued)

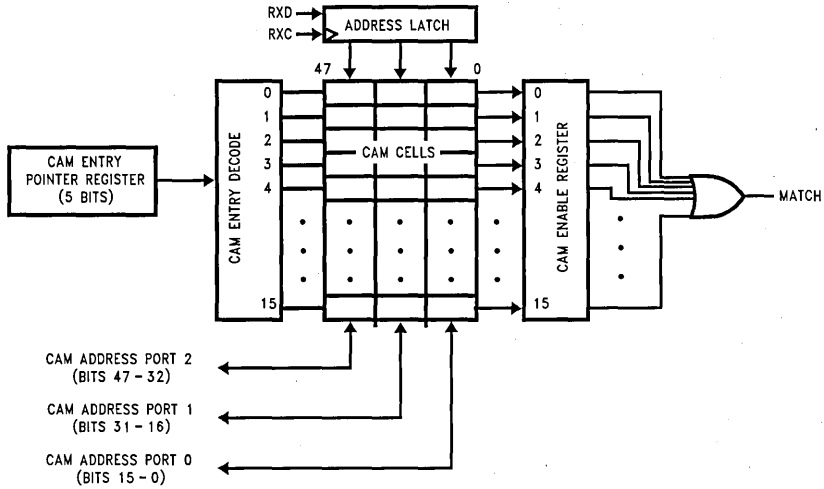


FIGURE 4-1. CAM Organization

TL/F/11722-21

the SONIC-16 is in software reset. The CDA resides in the same 64k byte block of memory as the Receive Resource Area (RRA) and contains descriptors for loading the CAM registers. These descriptors are contiguous and each descriptor consists of four 16-bit fields (*Figure 4-2*). The first field contains the value to be loaded into the CAM Entry Pointer and the remaining fields are for the three CAM Address Ports (see Section 4.3.10). In addition, there is one more field after the last descriptor containing the mask for the CAM Enable register. Each of the CAM descriptors are addressed by the CAM Descriptor Pointer (CDP) register.

After the system has initialized the CDA, it can issue the Load CAM command to program the SONIC-16 to read the CDA and load the CAM. The procedure for issuing the Load CAM command is as follows.

1. Initialize the Upper Receive Resource Address (URRA) register. Note that the CAM Descriptor Area must reside within the same 64k page as the Receive Resource Area. (See Section 4.3.9).

2. Initialize the CDA as described above.

3. Initialize the CAM Descriptor Count with the number of CAM descriptors. Note, only the lower 5 bits are used in this register. The other bits are don't cares. (See Section 4.3.10).

4. Initialize the CAM Descriptor Pointer to locate the first descriptor in the CDA. This register must be reloaded each time a new Load CAM command is issued.

5. Issue the Load CAM command (LCAM) in the Command register. (See Section 4.3.1).

If a transmission or reception is in progress, the CAM DMA function will not occur until these operations are complete. When the SONIC-16 completes the Load CAM command, the CDP register points to the next location after the CAM Enable field and the CDC equals zero. The SONIC-16 resets the LCAM bit in the Command register and sets the Load CAM Done (LCD) bit in the ISR.

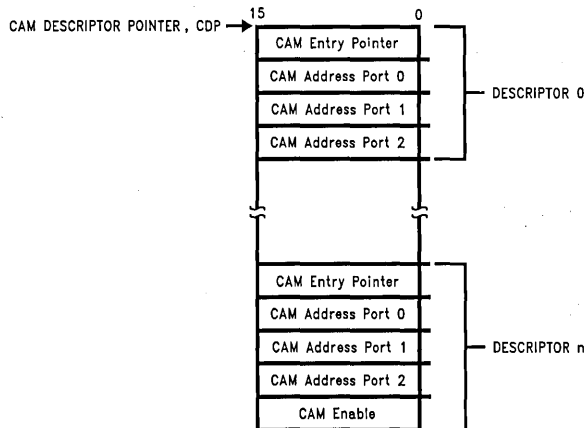


FIGURE 4-2. CAM Descriptor Area Format

TL/F/11722-22

### 4.0 SONIC-16 Registers (Continued)

RA <5:0>		15	0
Status and Control Registers	0h Command Register	Status and Control Fields	
	1 Data Configuration Register	Control Fields	
	2 Receive Control Register	Status and Control Fields	
	3 Transmit Control Register	Status and Control Fields	
	4 Interrupt Mask Register	Mask Fields	
	5 Interrupt Status Register	Status Fields	
Transmit Registers	3F Data Configuration Register 2	Control Fields	
	6 Upper Transmit Descriptor Address Register	Upper 16-bit Address Base	
	7 Current Transmit Descriptor Address Register	Lower 16-bit Address Offset	
Receive Registers	0D Upper Receive Descriptor Address Register	Upper 16-bit Address Base	
	0E Current Receive Descriptor Address Register	Lower 16-bit Address Offset	
	14 Upper Receive Resource Address Register	Upper 16-bit Address Base	
	15 Resource Start Address Register	Lower 16-bit Address Offset	
	16 Resource End Address Register	Lower 16-bit Address Offset	
	17 Resource Read Register	Lower 16-bit Address Offset	
	18 Resource Write Register	Lower 16-bit Address Offset	
	2B Receive Sequence Counter	Count Value	8 7
CAM Registers		4	
	21 CAM Entry Pointer	Pointer	
	22 CAM Address Port 2	Most Significant 16 bits of CAM Entry	
	23 CAM Address Port 1	Middle 16 bits of CAM Entry	
	24 CAM Address Port 0	Least Significant 16 bits of CAM Entry	
	25 CAM Enable Register	Mask Fields	
	26 CAM Descriptor Pointer	Lower 16-bit Address Offset	
		5	
Tally Counters	27 CAM Descriptor Count	Count Value	
	2C CRC Error Tally Counter	Count Value	
	2D Frame Alignment Error Tally	Count Value	
Watchdog Timer	2E Missed Packet Tally	Count Value	
	29 Watchdog Timer 0	Lower 16-bit Count Value	
	2A Watchdog Timer 1	Upper 16-bit Count Value	
	28 Silicon Revision Register	Chip Revision Number	

FIGURE 4-3. Register Programming Model

## 4.0 SONIC-16 Registers (Continued)

### 4.2 STATUS/CONTROL REGISTERS

This set of registers is used to convey status/control information to/from the host system and to control the operation of the SONIC-16. These registers are used for loading commands generated from the system, indicating transmit and receive status, buffering data to/from memory, and provid-

ing interrupt control. The registers are selected by asserting chip select to the SONIC-16 and providing the necessary address on register address pins RA5-RA0. Tables 4-1, 4-2, and 4-3 show the locations of all SONIC-16 registers and where information on the registers can be found in the data sheet.

TABLE 4-1. User Registers

RA5-RA0	Access	Register	Symbol	Description (section)
<b>COMMAND AND STATUS REGISTERS</b>				
00h	R/W	Command	CR	4.3.1
01 (Note 3)	R/W	Data Configuration	DCR	4.3.2
02	R/W	Receive Control	RCR	4.3.3
03	R/W	Transmit Control	TCR	4.3.4
04	R/W	Interrupt Mask	IMR	4.3.5
05	R/W	Interrupt Status	ISR	4.3.6
3F (Note 3)	R/W	Data Configuration 2	DCR2	4.3.7
<b>TRANSMIT REGISTERS</b>				
06	R/W	Upper Transmit Descriptor Address	UTDA	4.3.8, 3.4.4.1
07	R/W	Current Transmit Descriptor Address	CTDA	4.3.8, 3.5.3
<b>RECEIVE REGISTERS</b>				
0D	R/W	Upper Receive Descriptor Address	URDA	4.3.9, 3.4.4.1
0E	R/W	Current Receive Descriptor Address	CRDA	4.3.9, 3.4.4.3
13	R/W	End of Buffer Word Count	EOBC	4.3.9, 3.4.2
14	R/W	Upper Receive Resource Address	URRA	4.3.9, 3.4.4.1
15	R/W	Resource Start Address	RSA	4.3.9, 3.4.1
16	R/W	Resource End Address	REA	4.3.9, 3.4.1
17	R/W	Resource Read Pointer	RRP	4.3.9, 3.4.1
18	R/W	Resource Write Pointer	RWP	4.3.9, 3.4.1
2B	R/W	Receive Sequence Counter	RSC	4.3.9, 3.4.3.2
<b>CAM REGISTERS</b>				
21	R/W	CAM Entry Pointer	CEP	4.1, 4.3.10
22 (Note 1)	R	CAM Address Port 2	CAP2	4.1, 4.3.10
23 (Note 1)	R	CAM Address Port 1	CAP1	4.1, 4.3.10
24 (Note 1)	R	CAM Address Port 0	CAPO	4.1, 4.3.10
25 (Note 2)	R/W	CAM Enable	CE	4.1, 4.3.10
26	R/W	CAM Descriptor Pointer	CDP	4.1, 4.3.10
27	R/W	CAM Descriptor Count	CDC	4.1, 4.3.10
<b>TALLY COUNTERS</b>				
2C (Note 4)	R/W	CRC Error Tally	CRCT	4.3.11
2D (Note 4)	R/W	FAE Tally	FAET	4.3.11
2E (Note 4)	R/W	Missed Packet Tally	MPT	4.3.11

## 4.0 SONIC-16 Registers (Continued)

**TABLE 4-1. User Registers (Continued)**

RA5-RA0	Access	Register	Symbol	Description (section)
<b>WATCHDOG COUNTERS</b>				
29	R/W	Watchdog Timer 0	WT0	4.3.12
2A	R/W	Watchdog Timer 1	WT1	4.3.12
<b>SILICON REVISION</b>				
28	R	Silicon Revision	SR	4.3.13

**Note 1:** These registers can only be read when the SONIC-16 is in reset mode (RST bit in the CR is set). The SONIC-16 gives invalid data when these registers are read in non-reset mode.

**Note 2:** This register can only be written to when the SONIC-16 is in reset mode. This register is normally only loaded by the Load CAM command.

**Note 3:** The Data Configuration registers, DCR and DCR2, can only be written to when the SONIC-16 is in reset mode (RST bit in CR is set). Writing to these registers while not in reset mode does not alter the registers.

**Note 4:** The data written to these registers is inverted before being latched. That is, if a value of FFFFh is written, these registers will contain and read back the value of 0000h. Data is not inverted during a read operation.

**TABLE 4-2. Internal Use Registers (Users should not write to these registers)**

(RA5-RA0)	Access	Register	Symbol	Description (section)
<b>TRANSMIT REGISTERS</b>				
08 (Note 1)	R/W	Transmit Packet Size	TPS	3.5
09	R/W	Transmit Fragment Count	TFC	3.5
0A	R/W	Transmit Start Address 0	TSA0	3.5
0B	R/W	Transmit Start Address 1	TSA1	3.5
0C (Note 2)	R/W	Transmit Fragment Size	TFS	3.5
20	R/W	Temporary Transmit Descriptor Address	TTDA	3.5.4
2F	R	Maximum Deferral Timer	MDT	4.3.4

**RECEIVE REGISTERS**

0F	R/W	Current Receive Buffer Address 0	CRBA0	3.4.2, 3.4.4.2
10	R/W	Current Receive Buffer Address 1	CRBA1	3.4.2, 3.4.4.2
11	R/W	Remaining Buffer Word Count 0	RBWC0	3.4.2, 3.4.4.2
12	R/W	Remaining Buffer Word Count 1	RBWC1	3.4.2, 3.4.4.2
19	R/W	Temporary Receive Buffer Address 0	TRBA0	3.4.6.2
1A	R/W	Temporary Receive Buffer Address 1	TRBA1	3.4.6.2
1B	R/W	Temporary Buffer Word Count 0	TBWC0	3.4.6.2
1C	R/W	Temporary Buffer Word Count 1	TBWC1	3.4.6.2
1F	R/W	Last Link Field Address	LLFA	none

**ADDRESS GENERATORS**

1D	R/W	Address Generator 0	ADDR0	none
1E	R/W	Address Generator 1	ADDR1	none

**Note 1:** The data that is read from these registers is the inversion of what has been written to them.

**Note 2:** The value that is written to this register is shifted once.

**TABLE 4-3. National Factory Test Registers**

(RA5-RA0)	Access	Register	Symbol	Description (section)
30 • 3E	R/W	These registers are for factory use only. Users must not address these registers or improper SONIC-16 operation can occur.	none	none

## 4.0 SONIC-16 Registers (Continued)

### 4.3 REGISTER DESCRIPTION

#### 4.3.1 Command Register

(RA <5:0> = 0h)

This register (Figure 4-4) is used for issuing commands to the SONIC-16. These commands are issued by setting the corresponding bits for the function. For all bits, except for the RST bit, the SONIC-16 resets the bit after the command is completed. With the exception of RST, writing a "0" to any bit has no effect. Before any commands can be issued, the RST bit must first be reset to "0". This means that, if the RST bit is set, two writes to the Command Register are required to issue a command to the SONIC-16; one to clear the RST bit, and one to issue the command.

This register also controls the general purpose 32-bit Watchdog Timer. After the Watchdog Timer register has been loaded, it begins to decrement once the ST bit has been set to "1". An interrupt is issued when the count reaches zero if the Timer Complete interrupt is enabled in the IMR.

During hardware reset, bits 7, 4, and 2 are set to a "1"; all others are cleared. During software reset bits 9, 8, 1, and 0 are cleared and bits 7 and 2 are set to a "1"; all others are unaffected.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	LCAM	RRRA	RST	0	ST	STP	RXEN	RXDIS	TXP	HTX
						r/w	r/w	r/w		r/w	r/w	r/w	r/w	r/w	r/w

r = read only, r/w = read/write

FIGURE 4-4. Command Register

Field	Meaning
LCAM	LOAD CAM
RRRA	READ RRA
RST	SOFTWARE RESET
ST	START TIMER
STP	STOP TIMER
RXEN	RECEIVER ENABLE
RXDIS	RECEIVER DISABLE
TXP	TRANSMIT PACKET(S)
HTX	HALT TRANSMISSION

Bit	Description
15-10	Must be 0
9	<b>LCAM: LOAD CAM</b> Setting this bit causes the SONIC-16 to load the CAM with the descriptor that is pointed to by the CAM Descriptor Pointer register. <b>Note:</b> This bit must not be set during transmission (TXP is set). The SONIC-16 will lock up if both bits are set simultaneously.
8	<b>RRRA: READ RRA</b> Setting this bit causes the SONIC-16 to read the next RRA descriptor pointed to by the Resource Read Pointer (RRP) register. Generally this bit is only set during initialization. Setting this bit during normal operation can cause improper receive operation.
7	<b>RST: SOFTWARE RESET</b> Setting this bit resets all internal state machines. The CRC generator is disabled and the Tally counters are halted, but not cleared. The SONIC-16 becomes operational when this bit is reset to "0". A hardware reset sets this bit to "1". It must be reset to "0" before the SONIC-16 becomes operational.
6	Must be 0.
5	<b>ST: START TIMER</b> Setting this bit enables the general-purpose watchdog timer to begin counting or to resume counting after it has been halted. This bit is reset when the timer is halted (i.e., STP is set). Setting this bit resets STP.
4	<b>STP: STOP TIMER</b> Setting this bit halts the general-purpose watchdog timer and resets the ST bit. The timer resumes when the ST bit is set. This bit powers up as a "1". <b>Note:</b> Simultaneously setting bits ST and STP stops the timer.

## 4.0 SONIC-16 Registers (Continued)

### 4.3 REGISTER DESCRIPTION

#### 4.3.1 Command Register (Continued)

(RA<5:0> = 0h)

Bit	Description
3	<p><b>RXEN: RECEIVER ENABLE</b></p> <p>Setting this bit enables the receive buffer management engine to begin buffering data to memory. Setting this bit resets the RXDIS bit. Note: If this bit is set while the MAC unit is currently receiving a packet, both RXEN and RXDIS are set until the network goes inactive (i.e., the SONIC-16 will not start buffering in the middle of a packet being received).</p>
2	<p><b>RXDIS: RECEIVER DISABLE</b></p> <p>Setting this bit disables the receiver from buffering data to memory or the Receive FIFO. If this bit is set during the reception of a packet, the receiver is disabled only after the packet is processed. The RXEN bit is reset when the receiver is disabled. Tally counters remain active regardless of the state of this bit. Note: If this bit is set while the SONIC-16 is currently receiving a packet, both RXEN and RXDIS are set until the packet is fully received.</p>
1	<p><b>TXP: TRANSMIT PACKET(S)</b></p> <p>Setting this bit causes the SONIC-16 to transmit packets which have been set up in the Transmit Descriptor Area (TDA). The SONIC-16 loads its appropriate registers from the TDA, then begins transmission. The SONIC-16 clears this bit after any of the following conditions have occurred: (1) transmission had completed (i.e., after the SONIC-16 has detected EOL = 1), (2) the Halt Transmission command (HTX) has taken effect, or (3) a transmit abort condition has occurred. This condition occurs when any of the following bits in the TCR have been set: EXC, EXD, FU, or BCM.</p> <p><b>Note:</b> This bit must not be set if a Load CAM operation is in progress (LCAM is set). The SONIC-16 will lock up if both bits are set simultaneously.</p>
0	<p><b>HTX: HALT TRANSMISSION</b></p> <p>Setting this bit halts the transmit command after the current transmission has completed. TXP is reset after transmission has halted. The Current Transmit Descriptor Address (CTDA) register points to the last descriptor transmitted. The SONIC-16 samples this bit after writing to the Txpkt.status field.</p>

## 4.0 SONIC-16 Registers (Continued)

### 4.3.2 Data Configuration Register

(RA<5:0> = 1h)

This register (Figure 4-5) establishes the bus cycle options for reading/writing data to/from 16- or 32-bit memory systems.

During a hardware reset, bits 15 and 13 are cleared; all other bits are unaffected. (Because of this, the first thing the driver software does to the SONIC-16 should be to set up this register.) All bits are unaffected by a software reset. This register must only be accessed when the SONIC-16 is in reset mode (i.e., the RST bit is set in the Command register).

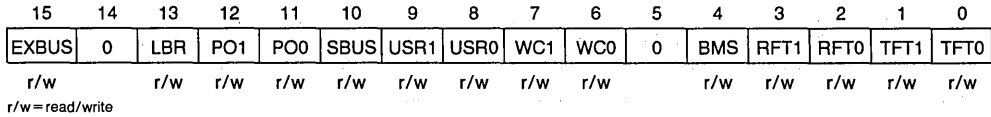


FIGURE 4-5. Data Configuration Register

Field	Meaning
EXBUS	EXTENDED BUS MODE
LBR	LATCHED BUS RETRY
PO0,PO1	PROGRAMMABLE OUTPUTS
SBUS	SYNCHRONOUS BUS MODE
USR0, USR1	USER DEFINABLE PINS
WC0, WC1	WAIT STATE CONTROL
DW	DATA WIDTH SELECT
BMS	BLOCK MODE SELECT FOR DMA
RFT0, RFT1	RECEIVE FIFO THRESHOLD
TFT0, TFT1	TRANSMIT FIFO THRESHOLD

Bit	Description
15	<p><b>EXBUS: EXTENDED BUS MODE</b></p> <p>Setting this bit enables the Extended Bus mode which enables the following:</p> <ol style="list-style-type: none"> <li>1) Extended Programmable Outputs, EXUSR &lt;3:0&gt;: This changes the TXD, LBK, RXC and RXD pins from the external ENDEC interface into four programmable user outputs, EXUSR &lt;3:0&gt; respectively, which are similar to USR &lt;1:0&gt;. These outputs are programmed with bits 15-12 in the DCR2 (see Section 4.3.7). On hardware reset, these four pins will be TRI-STATE® and will remain that way until the DCR is changed. If EXBUS is enabled, then these pins will remain TRI-STATE until the SONIC-16 becomes a bus master, at which time they will be driven according to the DCR2. If EXBUS is disabled, then these four pins work normally as external ENDEC interface pins.</li> <li>2) Synchronous Termination, STERM: This changes the TXC pin from the External ENDEC interface into a synchronous memory termination input for compatibility with Motorola style processors. This input is only useful when Asynchronous Bus mode is selected (bit 10 below is set to "0") and BMODE = 1 (Motorola mode). On hardware reset, this pin will be TRI-STATE and will remain that way until the DCR is changed. If EXBUS is enabled, this pin will remain TRI-STATE until the SONIC-16 becomes a bus master, at which time it will become the STERM input. If EXBUS is disabled, then this pin works normally as the TXC pin for the external ENDEC interface.</li> <li>3) Asynchronous Bus Retry: Causes BRT to be clocked in asynchronously off the falling edge of bus clock. This only applies, however, when the SONIC-16 is operating in asynchronous mode (bit 10 below is set to "0"). If EXBUS is not set, BRT is sampled synchronously off the rising edge of bus clock. (See Section 5.4.6.)</li> </ol>
14	Must be 0.
13	<p><b>LBR: LATCHED BUS RETRY</b></p> <p>The LBR bit controls the mode of operation of the <math>\overline{\text{BRT}}</math> signal (see pin description). It allows the BUS Retry operation to be latched or unlatched.</p> <p>0: Unlatched mode: The assertion of <math>\overline{\text{BRT}}</math> forces the SONIC-16 to finish the current DMA operation and get off the bus. The SONIC-16 will retry the operation when BRT is deserted.</p> <p>1: Latched mode: The assertion of <math>\overline{\text{BRT}}</math> forces the SONIC-16 to finish the current DMA operation as above, however, the SONIC-16 will not retry until BRT is deasserted and the BR bit in the ISR (see Section 4.3.6) has been reset. Hence, the mode has been latched on until the BR bit is cleared.</p> <p><b>Note:</b> Unless LBR is set to a "1", BRT must remain asserted at least until the SONIC-16 has gone idle. See Section 5.4.6 and the timing for Bus Retry in Section 7.0.</p>
12, 11	<p><b>PO1, PO0: PROGRAMMABLE OUTPUTS</b></p> <p>The PO1,PO0 bits individually control the USR1,0 pins respectively when SONIC-16 is a bus master (HLDA or <math>\overline{\text{BGACK}}</math> is active). When PO1/PO0 are set to a 1 the USR1/USR0 pins are high during bus master operations and when these bits are set to a 0 the USR1/USR0 pins are low during bus master operations.</p>



## 4.0 SONIC-16 Registers (Continued)

### 4.3.2 Data Configuration Register (Continued)

(RA<5:0> = 1h)

Bit	Description															
10	<p><b>SBUS: SYNCHRONOUS BUS MODE</b></p> <p>The SBUS bit is used to select the mode of system bus operation when SONIC-16 is a bus master. This bit selects the internal ready line to be either a synchronous or asynchronous input to SONIC-16 during block transfer DMA operations.</p> <p>0: Asynchronous mode. <math>\overline{RDY}_i</math> (BMODE = 0) or <math>\overline{DSACK0,1}</math> (BMODE = 1) are respectively internally synchronized at the falling edge of the bus clock (T2 of the DMA cycle). No setup or hold times need to be met with respect to this edge to guarantee proper bus operation.</p> <p>1: Synchronous mode. <math>\overline{RDY}_i</math> (BMODE = 0) and <math>\overline{DSACK0,1}</math> (BMODE = 1) must respectively meet the setup and hold times with respect to the rising edge of T1 or T2 to guarantee proper bus operation.</p>															
9, 8	<p><b>USR1,0: USER DEFINABLE PINS</b></p> <p>The USR1,0 bits report the level of the USR1,0 signal pins, respectively, after a chip hardware reset. If the USR1,0 signal pins are at a logical 1 (tied to V<sub>CC</sub>) during a hardware reset the USR1,0 bits are set to a 1. If the USR1,0 pins are at a logical 0 (tied to ground) during a hardware reset the USR1,0 bits are set to a 0. These bits are latched on the rising edge of <math>\overline{RST}</math>. Once set they remain set/reset until the next hardware reset.</p>															
7, 6	<p><b>WC1,0: WAIT STATE CONTROL</b></p> <p>These encoded bits determine the number of additional bus cycles (T2 states) that are added during each DMA cycle.</p> <table border="1"> <thead> <tr> <th>WC1</th> <th>WC0</th> <th>Bus Cycles Added</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>0</td> </tr> <tr> <td>0</td> <td>1</td> <td>1</td> </tr> <tr> <td>1</td> <td>0</td> <td>2</td> </tr> <tr> <td>1</td> <td>1</td> <td>3</td> </tr> </tbody> </table>	WC1	WC0	Bus Cycles Added	0	0	0	0	1	1	1	0	2	1	1	3
WC1	WC0	Bus Cycles Added														
0	0	0														
0	1	1														
1	0	2														
1	1	3														
5	<b>MUST BE 0.</b>															
4	<p><b>BMS: BLOCK MODE SELECT FOR DMA</b></p> <p>Determines how data is emptied or filled into the Receive or Transmit FIFO.</p> <p>0: Empty/fill mode: All DMA transfers continue until either the Receive FIFO has emptied or the Transmit FIFO has filled completely.</p> <p>1: Block mode: All DMA transfers continue until the programmed number of bytes (RFT0, RFT1 during reception or TFO, TF1 during transmission) have been transferred. (See note for TFT0, TFT1.)</p>															
3, 2	<p><b>RFT1,RFT0: RECEIVE FIFO THRESHOLD</b></p> <p>These encoded bits determine the number of words (or long words) that are written into the receive FIFO from the MAC unit before a receive DMA request occurs. (See Section 1.4.)</p> <table border="1"> <thead> <tr> <th>RFT1</th> <th>RFT0</th> <th>Threshold</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>2 words (4 bytes)</td> </tr> <tr> <td>0</td> <td>1</td> <td>4 words (8 bytes)</td> </tr> <tr> <td>1</td> <td>0</td> <td>8 words (16 bytes)</td> </tr> <tr> <td>1</td> <td>1</td> <td>12 words (24 bytes)</td> </tr> </tbody> </table> <p><b>Note:</b> In block mode (BMS bit = 1), the receive FIFO threshold sets the number of words (or long words) written to memory during a receive DMA block cycle.</p>	RFT1	RFT0	Threshold	0	0	2 words (4 bytes)	0	1	4 words (8 bytes)	1	0	8 words (16 bytes)	1	1	12 words (24 bytes)
RFT1	RFT0	Threshold														
0	0	2 words (4 bytes)														
0	1	4 words (8 bytes)														
1	0	8 words (16 bytes)														
1	1	12 words (24 bytes)														
1, 0	<p><b>TFT1,TFT0: TRANSMIT FIFO THRESHOLD</b></p> <p>These encoded bits determine the minimum number of words (or long words) the DMA section maintains in the transmit FIFO. A bus request occurs when the number of words drops below the transmit FIFO threshold. (See Section 1.4.)</p> <table border="1"> <thead> <tr> <th>TFT1</th> <th>TFT0</th> <th>Threshold</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>4 words (8 bytes)</td> </tr> <tr> <td>0</td> <td>1</td> <td>8 words (16 bytes)</td> </tr> <tr> <td>1</td> <td>0</td> <td>12 words (24 bytes)</td> </tr> <tr> <td>1</td> <td>1</td> <td>14 words (28 bytes)</td> </tr> </tbody> </table> <p><b>Note:</b> In block mode (BMS = 1), the number of bytes the SONIC-16 reads in a single DMA burst equals the transmit FIFO threshold value. If the number of words or long words needed to fill the FIFO is less than the threshold value, then only the number of reads required to fill the FIFO in a single DMA burst will be made. Typically, with the FIFO threshold value set to 12 or 14 words, the number of memory reads needed is less than the FIFO threshold value.</p>	TFT1	TFT0	Threshold	0	0	4 words (8 bytes)	0	1	8 words (16 bytes)	1	0	12 words (24 bytes)	1	1	14 words (28 bytes)
TFT1	TFT0	Threshold														
0	0	4 words (8 bytes)														
0	1	8 words (16 bytes)														
1	0	12 words (24 bytes)														
1	1	14 words (28 bytes)														

## 4.0 SONIC-16 Registers (Continued)

### 4.3.3 Receive Control Register

(RA<5:0> = 2h)

This register is used to filter incoming packets and provide status information of accepted packets (Figure 4-6). Setting any of bits 15–11 to a “1” enables the corresponding receive filter. If none of these bits are set, only packets which match the CAM Address registers are accepted. Bits 10 and 9 control the loopback operations.

After reception, bits 8–0 indicate status information about the accepted packet and are set to “1” when the corresponding condition is true. If the packet is accepted, all bits in the RCR are written into the RXpkt.status field. Bits 8–6 and 3–0 are cleared at the reception of the next packet.

This register is unaffected by a software reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR	RNT	BRD	PRO	AMC	LB1	LB0	MC	BC	LPKT	CRS	COL	CRCR	FAER	LBK	PRX
r/w	r/w	r/w	r/w	r/w	r/w	r/w	r	r	r	r	r	r	r	r	r

r = read only, r/w = read/write

FIGURE 4-6. Receive Control Register

Field	Meaning
ERR	ACCEPT PACKET WITH ERRORS
RNT	ACCEPT RUNT PACKETS
BRD	ACCEPT BROADCAST PACKETS
PRO	PHYSICAL PROMISCUOUS PACKETS
AMC	ACCEPT ALL MULTICAST PACKETS
LB0, LB1	LOOPBACK CONTROL
MC	MULTICAST PACKET RECEIVED
BC	BROADCAST PACKET RECEIVED
LPKT	LAST PACKET IN RBA
CRS	CARRIER SENSE ACTIVITY
COL	COLLISION ACTIVITY
CRCR	CRC ERROR
FAER	FRAME ALIGNMENT ERROR
LBK	LOOPBACK PACKET RECEIVED
PRX	PACKET RECEIVED OK

Bit	Description
15	<b>ERR: ACCEPT PACKET WITH CRC ERRORS OR COLLISIONS</b> 0: Reject all packets with CRC errors or when a collision occurs. 1: Accept packets with CRC errors and ignore collisions.
14	<b>RNT: ACCEPT RUNT PACKETS</b> 0: Normal address match mode. 1: Accept runt packets (packets less than 64 bytes in length). <b>Note:</b> A hardware reset clears this bit.
13	<b>BRD: ACCEPT BROADCAST PACKETS</b> 0: Normal address match mode. 1: Accept broadcast packets (packets with addresses that match the CAM are also accepted). <b>Note:</b> This bit is cleared upon hardware reset.
12	<b>PRO: PHYSICAL PROMISCUOUS MODE</b> Enable all Physical Address packets to be accepted. 0: Normal address match mode. 1: Promiscuous mode.
11	<b>AMC: ACCEPT ALL MULTICAST PACKETS</b> 0: Normal address match mode. 1: Enables all multicast packets to be accepted. Broadcast packets are also accepted regardless of the BRD bit. (Broadcast packets are a subset of multicast packets.)

## 4.0 SONIC-16 Registers (Continued)

### 4.3.3 Receive Control Register (Continued)

(RA<5:0> = 2h)

Bit	Description															
10, 9	<p><b>LB1, LB0: LOOPBACK CONTROL</b></p> <p>These encoded bits control loopback operations for MAC loopback, ENDEC loopback and Transceiver loopback. For proper operation, the CAM Address registers and Receive Control register must be initialized to accept the Destination address of the loopback packet (see Section 1.7).</p> <p><b>Note:</b> A hardware reset clears these bits.</p> <table border="1"> <thead> <tr> <th>LB1</th> <th>LB0</th> <th>Function</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>No loopback, normal operation</td> </tr> <tr> <td>0</td> <td>1</td> <td>MAC loopback</td> </tr> <tr> <td>1</td> <td>0</td> <td>ENDEC loopback</td> </tr> <tr> <td>1</td> <td>1</td> <td>Transceiver loopback</td> </tr> </tbody> </table>	LB1	LB0	Function	0	0	No loopback, normal operation	0	1	MAC loopback	1	0	ENDEC loopback	1	1	Transceiver loopback
LB1	LB0	Function														
0	0	No loopback, normal operation														
0	1	MAC loopback														
1	0	ENDEC loopback														
1	1	Transceiver loopback														
8	<p><b>MC: MULTICAST PACKET RECEIVED</b></p> <p>This bit is set when a packet is received with a Multicast Address.</p>															
7	<p><b>BC: BROADCAST PACKET RECEIVED</b></p> <p>This bit is set when a packet is received with a Broadcast Address.</p>															
6	<p><b>LPKT: LAST PACKET IN RBA</b></p> <p>This bit is set when the last packet is buffered into a Receive Buffer Area (RBA). The SONIC-16 detects this condition when its Remaining Buffer Word Count (RBWC0,1) register is less than the End Of Buffer Count (EOBC) register. (See Section 3.4.2.)</p>															
5	<p><b>CRS: CARRIER SENSE ACTIVITY</b></p> <p>Set when CRS is active. Indicates the presence of network activity.</p>															
4	<p><b>COL: COLLISION ACTIVITY</b></p> <p>Indicates that the packet received had a collision occur during reception.</p>															
3	<p><b>CRCR: CRC ERROR</b></p> <p>Indicates the packet contains a CRC error. If the packet also contains a Frame Alignment error, FAER will be set instead (see below).</p>															
2	<p><b>FAER: FRAME ALIGNMENT ERROR</b></p> <p>Indicates that the incoming packet was not correctly framed on an 8-bit boundary. Note: if no CRC errors have occurred, this bit is not set (i.e., this bit is only set when both a frame alignment and CRC errors occur).</p>															
1	<p><b>LBK: LOOPBACK PACKET RECEIVED</b></p> <p>Indicates that the SONIC-16 has successfully received a loopback packet.</p>															
0	<p><b>PRX: PACKET RECEIVED OK</b></p> <p>Indicates that a packet has been received without CRC, frame alignment, length (runt packet) errors or collisions.</p>															

## 4.0 SONIC-16 Registers (Continued)

### 4.3.4 Transmit Control Register (RA<5:0> = 3h)

This register is used to program the SONIC-16's transmit actions and provide status information after a packet has been transmitted (Figure 4-7). At the beginning of transmission, bits 15, 14, 13 and 12 from the TXpkt.config field are loaded into the TCR to configure the various transmit modes (see section 3.5.1.1). When the transmission ends, bits 10-0 indicate status information and are set to a "1" when the corresponding condition is true. These bits, along with the number of collisions information, are written into the TXpkt.status field at the end of transmission (see section 3.5.1.2). Bits 9 and 5 are cleared after the TXpkt.status field has been written. Bits 10, 7, 6, and 1 are cleared at the commencement of the next transmission while bit 8 is set at this time.

A hardware reset sets bits 8 and 1 to a "1". This register is unaffected by a software reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PINTR	POWC	CRCI	EXDIS	0	EXD	DEF	NCRS	CRSL	EXC	OWC	0	PMB	FU	BCM	PTX
r/w	r/w	r/w	r/w		r	r	r	r	r	r		r	r	r	r

r = read only, r/w = read/write

FIGURE 4-7. Transmit Control Register

Field	Meaning
PINTR	PROGRAMMABLE INTERRUPT
POWC	PROGRAMMED OUT OF WINDOW COLLISION TIMER
CRCI	CRC INHIBIT
EXDIS	DISABLE EXCESSIVE DEFERRAL TIMER
EXD	EXCESSIVE DEFERRAL
DEF	DEFERRED TRANSMISSION
NCRS	NO CRS
CRSL	CRS LOST
EXC	EXCESSIVE COLLISIONS
OWC	OUT OF WINDOW COLLISION
PMB	PACKET MONITORED BAD
FU	FIFO UNDERRUN
BCM	BYTE COUNT MISMATCH
PTX	PACKET TRANSMITTED OK

Bit	Description
15	<p><b>PINTR: PROGRAMMABLE INTERRUPT</b></p> <p>This bit allows transmit interrupts to be generated under software control. The SONIC-16 will issue an interrupt (PINT in the Interrupt Status Register) immediately after reading a TDA and detecting that PINTR is set in the TXpkt.config field.</p> <p><b>Note:</b> In order for PINTR to operate properly, it must be set and reset in the TXpkt.config field by alternating TDAs. This is necessary because after PINT has been issued in the ISR, PINTR in the Transmit Control Register must be cleared before it is set again in order to have the interrupt issued for another packet. The only effective way to do this is to set PINTR to a 1 no more often than every other packet.</p>
14	<p><b>POWC: PROGRAM "OUT OF WINDOW COLLISION" TIMER</b></p> <p>This bit programs when the out of window collision timer begins.</p> <p>0: timer begins after the Start of Frame Delimiter (SFD).</p> <p>1: timer begins after the first bit of preamble.</p>
13	<p><b>CRCI: CRC INHIBIT</b></p> <p>0: transmit packet with 4-byte FCS field</p> <p>1: transmit packet without 4-byte FCS field</p>
12	<p><b>EXDIS: DISABLE EXCESSIVE DEFERRAL TIMER:</b></p> <p>0: excessive deferral timer enabled</p> <p>1: excessive deferral timer disabled</p>
11	Must be 0.
10	<p><b>EXD: EXCESSIVE DEFERRAL</b></p> <p>Indicates that the SONIC-16 has been deferring for 3.2 ms. The transmission is aborted if the excessive deferral timer is enabled (i.e. EXDIS is reset). This bit can only be set if the excessive deferral timer is enabled.</p>

## 4.0 SONIC-16 Registers (Continued)

### 4.3.4 Transmit Control Register (Continued)

(RA<5:0> = 3h)

Bit	Description
9	<p><b>DEF: DEFERRED TRANSMISSION</b></p> <p>Indicates that the SONIC-16 has deferred its transmission during the first attempt. If subsequent collisions occur, this bit is reset. This bit is cleared after the TXpkt.status field is written in the TDA.</p>
8	<p><b>NCRS: NO CRS</b></p> <p>Indicates that Carrier Sense (CRS) was not present during transmission. CRS is monitored from the beginning of the Start of Frame Delimiter to the last byte transmitted. The transmission will not be aborted. This bit is set at the start of preamble and is reset if CRS is detected. Hence, if CRS is never detected throughout the entire transmission of the packet, this bit will remain set.</p> <p><b>Note:</b> NCRS will always remain set in MAC loopback.</p>
7	<p><b>CRSL: CRS LOST</b></p> <p>Indicates that CRS has gone low or has not been present during transmission. CRS is monitored from the beginning of the Start of Frame Delimiter to the last byte transmitted. The transmission will not be aborted.</p> <p><b>Note:</b> if CRS was never present, both NCRS and CRSL will be set simultaneously. Also, CRSL will always be set in MAC loopback.</p>
6	<p><b>EXC: EXCESSIVE COLLISIONS</b></p> <p>Indicates that 16 collisions have occurred. The transmission is aborted.</p>
5	<p><b>OWC: OUT OF WINDOW COLLISION</b></p> <p>Indicates that an illegal collision has occurred after 51.2 <math>\mu</math>s (one slot time) from either the first bit of preamble or from SFD depending upon the POWC bit. The transmission backs off as in a normal transmission. This bit is cleared after the TXpkt.status field is written in the TDA.</p>
4	Must be 0.
3	<p><b>PMB: PACKET MONITORED BAD</b></p> <p>This bit is set, if after the receive unit has monitored the transmitted packet, the CRC has been calculated as invalid, a frame alignment error occurred or the Source Address does not match any of the CAM address registers.</p> <p><b>Note 1:</b> The SONIC-16's CRC checker is active during transmission.</p> <p><b>Note 2:</b> If CRC has been inhibited for transmissions (CRCI is set), this bit will always be low. This is true regardless of Frame Alignment or Source Address mismatch errors.</p> <p><b>Note 3:</b> If a Receive FIFO overrun has occurred, the transmitted packet is not monitored completely. Thus, if PMB is set along with the RFO bit in the ISR, then PMB has no meaning. The packet must be completely received before PMB has meaning.</p>
2	<p><b>FU: FIFO UNDERRUN</b></p> <p>Indicates that the SONIC-16 has not been able to access the bus before the FIFO has emptied. This condition occurs from excessive bus latency and/or slow bus clock. The transmission is aborted. (See section 1.4.2.)</p>
1	<p><b>BCM: BYTE COUNT MISMATCH</b></p> <p>This bit is set when the SONIC-16 detects that the TXpkt.pkt_size field is not equal to the sum of the TXpkt.frag_size field(s). Transmission is aborted.</p>
0	<p><b>PTX: PACKET TRANSMITTED OK</b></p> <p>Indicates that a packet has been transmitted without the following errors:</p> <ul style="list-style-type: none"> <li>—Excessive Collisions (EXC)</li> <li>—Excessive Deferral (EXD)</li> <li>—FIFO Underrun (FU)</li> <li>—Byte Count Mismatch (BCM)</li> </ul>

## 4.0 SONIC-16 Registers (Continued)

### 4.3.5 Interrupt Mask Register

(RA < 5:0 > = 4h)

This register masks the interrupts that can be generated from the ISR (Figure 4-8). Writing a "1" to the bit enables the corresponding interrupt. During a hardware reset, all mask bits are cleared.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	BREN	HBLEN	LCDEN	PINTEN	PRXEN	PTXEN	TXEREN	TCEN	RDEEN	RBEEN	RBAEEN	CRCEN	FAEEN	MPEN	RFOEN
	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

r/w = read/write

**FIGURE 4-8. Interrupt Mask Register**

Field	Meaning
BREN	BUS RETRY OCCURRED ENABLE
HBLEN	HEARTBEAT LOST ENABLE
LCDEN	LOAD CAM DONE INTERRUPT ENABLE
PINTEN	PROGRAMMABLE INTERRUPT ENABLE
PRXEN	PACKET RECEIVED ENABLE
PTXEN	PACKET TRANSMITTED OK ENABLE
TXEREN	TRANSMIT ERROR ENABLE
TCEN	TIMER COMPLETE ENABLE
RDEEN	RECEIVE DESCRIPTORS ENABLE
RBEEN	RECEIVE BUFFERS EXHAUSTED ENABLE
RBAEEN	RECEIVE BUFFER AREA EXCEEDED ENABLE
CRCEN	CRC TALLY COUNTER WARNING ENABLE
FAEEN	FAE TALLY COUNTER WARNING ENABLE
MPEN	MP TALLY COUNTER WARNING ENABLE
RFOEN	RECEIVE FIFO OVERRUN ENABLE

Bit	Description
15	Must be 0.
14	<b>BREN: BUS RETRY OCCURRED enable:</b> 0: disable 1: enables interrupts when a Bus Retry operation is requested.
13	<b>HBLEN: HEARTBEAT LOST enable:</b> 0: disable 1: enables interrupts when a heartbeat lost condition occurs.
12	<b>LCDEN: LOAD CAM DONE INTERRUPT enable:</b> 0: disable 1: enables interrupts when the Load CAM command has finished.
11	<b>PINTEN: PROGRAMMABLE INTERRUPT enable:</b> 0: disable 1: enables programmable interrupts to occur when the PINTR bit the TXpkt.config field is set to a "1".
10	<b>PRXEN: PACKET RECEIVED enable:</b> 0: disable 1: enables interrupts for packets accepted.
9	<b>PTXEN: PACKET TRANSMITTED OK enable:</b> 0: disable 1: enables interrupts for transmit completions.
8	<b>TXEREN: TRANSMIT ERROR enable:</b> 0: disable 1: enables interrupts for packets transmitted with error.

## 4.0 SONIC-16 Registers (Continued)

### 4.3.5 Interrupt Mask Register (Continued)

(RA<5:0> = 4h)

Bit	Description
7	<b>TCEN: GENERAL PURPOSE TIMER COMPLETE enable:</b> 0: disable 1: enables interrupts when the general purpose timer has rolled over from 0000 0000h to FFFF FFFFh.
6	<b>RDEEN: RECEIVE DESCRIPTORS EXHAUSTED enable:</b> 0: disable 1: enables interrupts when all receive descriptors in the RDA have been exhausted.
5	<b>RBEEN: RECEIVE BUFFERS EXHAUSTED enable:</b> 0: disable 1: enables interrupts when all resource descriptors in the RRA have been exhausted.
4	<b>RBAEEN: RECEIVE BUFFER AREA EXCEEDED enable:</b> 0: disable 1: enables interrupts when the SONIC-16 attempts to buffer data beyond the end of the Receive Buffer Area.
3	<b>CRCCEN: CRC TALLY COUNTER WARNING enable:</b> 0: disable 1: enables interrupts when the CRC tally counter has rolled over from FFFFh to 0000h.
2	<b>FAEEN: FRAME ALIGNMENT ERROR (FAE) TALLY COUNTER WARNING enable:</b> 0: disable 1: enables interrupts when the FAE tally counter rolled over from FFFFh to 0000h.
1	<b>MPEN: MISSED PACKET (MP) TALLY COUNTER WARNING enable:</b> 0: disable 1: enables interrupts when the MP tally counter has rolled over from FFFFh to 0000h.
0	<b>RFOEN: RECEIVE FIFO OVERRUN enable:</b> 0: disable 1: enables interrupts when the receive FIFO has overrun.

## 4.0 SONIC-16 Registers (Continued)

### 4.3.6 Interrupt Status Register

(RA<5:0> = 5h)

This register (Figure 4-9) indicates the source of an interrupt when the INT pin goes active. Enabling the corresponding bits in the IMR allows bits in this register to produce an interrupt. When an interrupt is active, one or more bits in this register are set to a "1". A bit is cleared by writing "1" to it. Writing a "0" to any bit has no effect.

This register is cleared by a hardware reset and unaffected by a software reset.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	BR	HBL	LCD	PINT	PKTRX	PTDN	TXER	TC	RDE	RBE	RBAE	CRC	FAE	MP	RFO
	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w	r/w

r/w = read/write

FIGURE 4-9. Interrupt Status Register

Field	Meaning
BR	BUS RETRY OCCURRED
HBL	CD HEARTBEAT LOST
LCD	LOAD CAM DONE
PINT	PROGRAMMABLE INTERRUPT
PKTRX	PACKET RECEIVED
TXDN	TRANSMISSION DONE
TXER	TRANSMIT ERROR
TC	TIMER COMPLETE
RDE	RECEIVE DISCRIPTORS EXHAUSTED
RBE	RECEIVE BUFFERS EXHAUSTED
RBAE	RECEIVE BUFFER AREA EXCEEDED
CRC	CRC TALLY COUNTER ROLLOVER
FAE	FRAME ALIGNMENT ERROR
MP	MISSED PACKET COUNTER ROLLOVER
RFO	RECEIVE FIFO OVERRUN

Bit	Description
15	Must be 0.
14	<b>BR: BUS RETRY OCCURRED</b> Indicates that a Bus Retry (BRT) operation has occurred. In Latched Bus Retry mode (LBR in the DCR), BR will only be set when the SONIC-16 is a bus master. Before the SONIC-16 will continue any DMA operations, BR must be cleared. In Unlatched mode, the BR bit should be cleared also, but the SONIC-16 will not wait for BR to be cleared before requesting the bus again and continuing its DMA operations. (See sections 4.3.2 and 5.4.6 for more information on Bus Retry).
13	<b>HBL: CD HEARTBEAT LOST</b> If the transceiver fails to provide a collision pulse (heart beat) during the first 6.4 μs of the Interframe Gap after transmission, this bit is set.
12	<b>LCD: LOAD CAM DONE</b> Indicates that the Load CAM command has finished writing to all programmed locations in the CAM. (See section 4.1.1.)
11	<b>PINT: PROGRAMMED INTERRUPT</b> Indicates that upon reading the TXpkt.config field, the SONIC-16 has detected the PINTR bit to be set. (See section 4.3.4.)
10	<b>PKTRX: PACKET RECEIVED</b> Indicates that a packet has been received and been buffered to memory. This bit is set after the RXpkt.seq__no field is written to memory.
9	<b>TXDN: TRANSMISSION DONE</b> Indicates that either (1) there are no remaining packets to be transmitted in the Transmit Descriptor Area (i.e., the EOL bit has been detected as a "1"), (2) the Halt Transmit command has been given (HTX bit in CR is set to a "1"), or (3) a transmit abort condition has occurred. This condition occurs when any of following bits in the TCR are set: BCM, EXC, FU, or EXD. This bit is set after the TXpkt.status field has been written to.



## 4.0 SONIC-16 Registers (Continued)

### 4.3.6 Interrupt Status Register (Continued)

(RA <5:0> = 5h)

Bit	Description
8	<p><b>TXER: TRANSMIT ERROR</b>            Indicates that a packet has been transmitted with at least one of the following errors.</p> <ul style="list-style-type: none"> <li>—Byte count mismatch (BCM)</li> <li>—Excessive collisions (EXC)</li> <li>—FIFO underrun (FU)</li> <li>—Excessive deferral (EXD)</li> </ul> <p>The TXpkt.status field reveals the cause of the error(s).</p>
7	<p><b>TC: GENERAL PURPOSE TIMER COMPLETE</b>            Indicates that the timer has rolled over from 0000 0000h to FFFF FFFFh. (See section 4.3.12.)</p>
6	<p><b>RDE: RECEIVE DESCRIPTORS EXHAUSTED</b>            Indicates that all receive packet descriptors in the RDA have been exhausted. This bit is set when the SONIC-16 detects EOL = 1. (See section 3.4.7.)</p>
5	<p><b>RBE: RECEIVE BUFFER EXHAUSTED</b>            Indicates that the SONIC-16 has detected the Resource Read Pointer (RRP) is equal to the Resource Write Pointer (RWP). This bit is set after the last field is read from the resource area. (See section 3.4.7.)</p> <p><b>Note 1:</b> This bit will be set as the SONIC-16 finishes using the second to last receive buffer and reads the last RRA descriptor. This gives the system an early warning of impending no resources.</p> <p><b>Note 2:</b> The SONIC-16 will stop reception of packets when the last RBA has been used and will not continue reception until additional receive buffers have been added (i.e., RWP is incremented beyond RRP) and this bit has been reset.</p> <p><b>Note 3:</b> If additional buffers have been added, resetting this bit causes the SONIC-16 to read the next resource descriptor pointed to by the RRP in the Receive Resource Area. Note that resetting this bit under this condition is similar to issuing the Read RRA command (setting the RRRR bit in the Command Register). This bit should never be reset until after the additional resources have been added to the RRA.</p>
4	<p><b>RBAE: RECEIVE BUFFER AREA EXCEEDED</b>            Indicates that during reception, the SONIC-16 has reached the end of the Receive Buffer Area. Reception is aborted and the SONIC-16 fetches the next available resource descriptors in the RRA. The buffer space is not re-used and an RDA is not set up for the truncated packet (see section 3.4.7).</p>
3	<p><b>CRC: CRC TALLY COUNTER ROLLOVER</b>            Indicates that the tally counter has rolled over from FFFFh to 0000h. (See section 4.3.11.)</p>
2	<p><b>FAE: FRAME ALIGNMENT ERROR (FAE) TALLY COUNTER ROLLOVER</b>            Indicates that the FAE tally counter has rolled over from FFFFh to 0000h. (See section 4.3.11.)</p>
1	<p><b>MP: MISSED PACKET (MP) COUNTER ROLLOVER</b>            Indicates that the MP tally counter has rolled over from FFFFh to 0000h. (See section 4.3.11.)</p>
0	<p><b>RFO: RECEIVE FIFO OVERRUN</b>            Indicates that the SONIC-16 has been unable to access the bus before the receive FIFO has filled from the network. This condition is due to excessively long bus latency and/or slow bus clock. Note that FIFO underruns are indicated in the TCR. (See section 1.4.1.)</p>

## 4.0 SONIC-16 Registers (Continued)

### 4.3.7 Data Configuration Register 2

(RA<5:0> = 3Fh)

This register (Figure 4-10) is for enabling the extended bus interface options.

A hardware reset will set all bits in this register to "0" except for the Extended Programmable Outputs which are unknown until written to and bits 5 to 11 which must always be written with zeroes, but are "don't cares" when read. A software reset will not affect any bits in this register. This register should only be written to when the SONIC-16 is in software reset (the RST bit in the Command Register is set).

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
EXPO3	EXPO2	EXPO1	EXPO0	0	0	0	0	0	0	0	PH	0	PCM	PCNM	RJCM
r/w	r/w	r/w	r/w								r/w		r/w	r/w	r/w

FIGURE 4-10. Data Configuration Register 2

Field	Meaning
EXPO3..0	EXTENDED PROGRAMMABLE OUTPUTS
PH	PROGRAM HOLD
PCM	PACKET COMPRESS WHEN MATCHED
PCNM	PACKET COMPRESS WHEN NOT MATCHED
RJCM	REJECT ON CAM MATCH

Bit	Description
15-12	<p><b>EXPO&lt;3:0&gt; EXTENDED PROGRAMMABLE OUTPUTS</b></p> <p>These bits program the level of the Extended User outputs (EXUSR&lt;3:0&gt;) when the SONIC-16 is a bus master. Writing a "1" to any of these bits programs a high level to the corresponding output. Writing a "0" to any of these bits programs a low level to the corresponding output. EXUSR&lt;3:0&gt; are similar to USR&lt;1:0&gt; except that EXUSR&lt;3:0&gt; are only available when the Extended Bus mode is selected (bit 15 in the DCR is set to "1", see Section 4.3.2).</p>
11-5	Must be written with zeroes.
4	<p><b>PH: PROGRAM HOLD</b></p> <p>When this bit is set to "0", the HOLD request output is asserted/deasserted from the falling edge of bus clock. If this bit is set to "1", HOLD will be asserted/deasserted 1/2 clock later on the rising edge of bus clock.</p>
3	Must be zero.
2	<p><b>PCM: PACKET COMPRESS WHEN MATCHED</b></p> <p>When this bit is set to a "1" (and the PCNM bit is reset to a "0"), the <math>\overline{\text{PCOMP}}</math> output will be asserted if the destination address of the packet being received matches one of the entries in the CAM (Content Addressable Memory). This bit, along with PCNM, is used with the Management Bus of the DP83950, Repeater Interface Controller (RIC). See the DP83950 datasheet for more details on the RIC Management Bus. This mode is also called the Managed Bridge Mode.</p> <p><b>Note 1:</b> Setting PCNM and PCM to "1" at the same time is not allowed.</p> <p><b>Note 2:</b> If PCNM and PCM are both "0", the <math>\overline{\text{PCOMP}}</math> output will remain TRI-STATE until PCNM or PCM are changed.</p>
1	<p><b>PCNM: COMPRESS WHEN NOT MATCHED</b></p> <p>When this bit is set to a "1" (and the PCM bit is set to "0"), the <math>\overline{\text{PCOMP}}</math> output will be asserted if the destination address of the packet does not match one of the entries in the CAM. See the PCM bit above. This mode is also called the Managed Hub Mode.</p> <p><b>Note:</b> <math>\overline{\text{PCOMP}}</math> will not be asserted if the destination address is a broadcast address. This is true regardless of the state of the BRD bit in the Receive Control Register.</p>
0	<p><b>RJCM: REJECT ON CAM MATCH</b></p> <p>When this bit is set to "1", the SONIC-16 will reject a packet on a CAM match. Setting RJCM to "0" causes the SONIC-16 to operate normally by accepting packets on a CAM match. Setting this mode is useful for a small bridge with a limited number of nodes attached to it. RJCM only affects the CAM, though. Setting RJCM will not invert the function of the BRD, PRO or AMC bits (to accept broadcast, all physical or multicast packets respectively) in the Receive Control Register (see Section 4.3.3). This means, for example, that it is not possible to set RJCM and BRD to reject all broadcast packets. If RJCM and BRD are set at the same time, however, all broadcast packets will be accepted, but any packets that have a destination address that matches an address in the CAM will be rejected.</p>

## 4.0 SONIC-16 Registers (Continued)

### 4.3.8 Transmit Registers

The transmit registers described in this section are part of the User Register set. The UTDA and CTDA must be initialized prior to issuing the transmit command (setting the TXP bit) in the Command register.

#### Upper Transmit Descriptor Address Register (UTDA):

This register contains the upper address bits (A<23:16>) for accessing the transmit descriptor area (TDA) and is concatenated with the contents of the CTDA when the SONIC-16 accesses the TDA in system memory. The TDA can be as large as 32k words and can be located anywhere in system memory. This register is unaffected by a hardware or software reset.

#### Current Transmit Descriptor Address Register (CTDA):

The 16-bit CTDA register contains the lower address bits (A<15:1>) of the 23-bit transmit descriptor address. During initialization this register must be programmed with the lower address bits of the transmit descriptor. The SONIC-16 concatenates the contents of this register with the contents of the UTDA to point to the transmit descriptor. Bit 0 of this register is the End of List (EOL) bit and is used to denote the end of the list. This register is unaffected by a hardware or software reset.

### 4.3.9 Receive Registers

The receive registers described in this section are part of the User Register set. A software reset has no effect on these registers and a hardware reset only affects the EOBC and RSC registers. The receive registers must be initialized prior to issuing the receive command (setting the RXEN bit) in the Command register.

#### Upper Receive Descriptor Address Register (URDA):

This register contains the upper address bits (A<23:16>) for accessing the receive descriptor area (RDA) and is concatenated with the contents of the CRDA when the SONIC-16 accesses the RDA in system memory. The RDA can be as large as 32k words and can be located anywhere in system memory. This register is unaffected by a hardware or software reset.

#### Current Receive Descriptor Address Register (CRDA):

The CRDA is a 16-bit read/write register used to locate the received packet descriptor block within the RDA. It contains the lower address bits (A<15:1>). The SONIC-16 concatenates the contents of the CRDA with the contents of the URDA to form the complete 23-bit address. The resulting 23-bit address points to the first field of the descriptor block. Bit 0 of this register is the End of List (EOL) bit and is used to denote the end of the list. This register is unaffected by a hardware or software reset.

**End of Buffer Word Count Register (EOBC):** The SONIC-16 uses the contents of this register to determine where to place the next packet. At the end of packet reception, the SONIC-16 compares the contents of the EOBC register with the contents of the Remaining Buffer Word Count registers (RBWC0,1) to determine whether: (1) to place the next packet in the same RBA or (2) to place the next packet in another RBA. If the EOBC is less than or equal to the remaining number of words in the RBA after a packet is received (i.e.,  $EOBC \leq RBWC0,1$ ), the SONIC-16 buffers the next packet in the same RBA. If the EOBC is greater than

the remaining number of words in the RBA after a packet is received (i.e.,  $EOBC > RBWC0,1$ ), the Last Packet in RBA bit, LPKT in the Receive Control Register, section 4.3.3, is set and the SONIC-16 fetches the next resource descriptor. Hence, the next packet received will be buffered in a new RBA. A hardware reset sets this register to 02F8H (760 words or 1520 bytes). See sections 3.4.2 and 3.4.4.4 for more information about using EOBC.

#### Upper Receive Resource Address Register (URRA):

The URRA is a 16-bit read/write register. It is programmed with the base address of the receive resource area (RRA). This 8-bit upper address value (A<23:16>) locates the receive resource area in system memory. SONIC-16 uses the URRA register when accessing the receive descriptors within the RRA by concatenating the lower address value from one of four receive resource registers (RSA, REA, RWP, or RRP).

**Resource Start Address Register (RSA):** The RSA is a 15-bit read/write register. The LSB is not used and always reads back as a 0. The RSA is programmed with the lower 15-bit address (A<15:1>) of the starting address of the receive resource area. SONIC-16 concatenates the contents of this register with the contents of the URRA to form the complete 23-bit address.

**Resource End Address Register (REA):** The REA is a 15-bit read/write register. The LSB is not used and always reads back as a 0. The REA is programmed with the lower 15-bit address (A<15:1>) of the ending address of the receive resource area. SONIC-16 concatenates the contents of this register with the contents of the URRA to form the complete 23-bit address.

**Resource Read Pointer Register (RRP):** The RRP is a 15-bit read/write register. The LSB is not used and always reads back as a 0. The RRP is programmed with the lower 15-bit address (A<15:1>) of the first field of the next descriptor the SONIC-16 will read. SONIC-16 concatenates the contents of this register with the contents of the URRA to form the complete 23-bit address.

**Resource Write Pointer Register (RWP):** The RWP is a 15-bit read/write register. The LSB is not used and always reads back as a 0. The RWP is programmed with the lower 15-bit address (A<15:1>) of the next available location the system can add a descriptor. SONIC-16 concatenates the contents of this register with the contents of the URRA to form the complete 23-bit address.

**Receive Sequence Counter Register (RSC):** This is a 16-bit read/write register containing two fields. The SONIC-16 uses this register to provide status information on the number of packets within a RBA and the number of RBAs. The RSC register contains two 8-bit (modulo 256) counters. After each packet is received the packet sequence number is incremented. The SONIC-16 maintains a single sequence number for each RBA. When the SONIC-16 uses the next RBA, the packet sequence number is reset to zero and the RBA sequence number is incremented. This register is reset to 0 by a hardware reset or by writing zero to it. A software reset has no affect.

15	8	7	0
RBA Sequence Number (Modulo 256)		Packet Sequence Number (Modulo 256)	

## 4.0 SONIC-16 Registers (Continued)

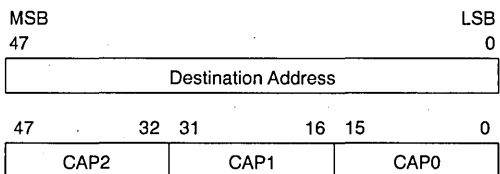
### 4.3.10 CAM Registers

The CAM registers described in this section are part of the User Register set. They are used to program the Content Addressable Memory (CAM) entries that provide address filtering of packets. These registers, except for the CAM Enable register, are unaffected by a hardware or software reset.

**CAM Entry Pointer Register (CEP):** The CEP is a 4-bit register used by SONIC-16 to select one of the sixteen CAM entries. SONIC-16 uses the least significant 4-bits of this register. The value of 0h points to the first CAM entry and the value of Fh points to the last entry.

**CAM Address Port 2, 1, 0 Registers (CAP2, CAP1, CAP0):** Each CAP is a 16-bit read-only register used to access the CAM cells. Each CAM cell is 16-bits wide and contains one third of the 48-bit CAM entry which is used by the SONIC-16 for address filtering. The CAP2 register is used to access the upper bits (<47:32>), CAP1 the middle bits (<31:16>) and CAP0 the lower bits (<15:0>) of the CAM entry. Given the physical address 10:20:30:40:50:60, which is made up of 6 octets or bytes, where 10h is the least significant byte and 60h is the most significant byte (10h would be the first byte received from the network and 60h would be the last), CAP0 would be loaded with 2010h, CAP1 with 4030h and CAP2 with 6050h.

To read a CAM entry, the user first places the SONIC-16 in software reset (set the RST bit in the Command register), programs the CEP register to select one of sixteen CAM entries, then reads CAP2, CAP1, and CAP0 to obtain the complete 48-bit entry. The user can not write to the CAM entries directly. Instead, the user programs the CAM descriptor area in system memory (see section 4.1.1), then issues the Load CAM command (setting LCAM bit in the Command register). This causes the SONIC-16 to read the descriptors from memory and loads the corresponding CAM entry through CAP2-0.



**CAM Enable Register (CE):** The CE is a 16-bit read/write register used to mask out or enable individual CAM entries. Each register bit position corresponds to a CAM entry. When a register bit is set to a "1" the corresponding CAM entry is enabled. When "0" the entry is disabled. This register is unaffected by a software reset and cleared to zero (disabling all entries) during a hardware reset. Under normal operations the user does not access this register. Instead the user sets up this register through the last entry in the CAM descriptor area. The SONIC-16 loads the CE register during execution of the LCAM Command.

**CAM Descriptor Pointer Register (CDP):** The CDP is a 15-bit read/write register. The LSB is unused and always reads back as 0. The CDP is programmed with the lower

address (A<15:1>) of the first field of the CAM descriptor block in the CAM descriptor area (CDA) of system memory. SONIC-16 uses the contents of the CDP register when accessing the CAM descriptors. This register must be programmed by the user before issuing the LCAM command. During execution of the LCAM Command SONIC-16 concatenates the contents of this register with the contents of the URRR register to form the complete 23-bit address. During the Load CAM operation this register is incremented to address the fields in the CDA. After the Load Command completes this register points to the next location after the CAM Descriptor Area.

**CAM Descriptor Count Register (CDC):** The CDC is a 5-bit read/write register. It is programmed with the number of CAM descriptor blocks in the CAM descriptor area. This register must be programmed by the user before issuing the LCAM command. SONIC-16 uses the value in this register to determine how many entries to place in the CAM during execution of the LCAM command. During LCAM execution SONIC-16 decrements this register each time it reads a descriptor block. When the CDC decrements to zero SONIC-16 terminates the LCAM execution. Since the CDC register is programmed with the number of CAM descriptor blocks in the CAM Descriptor Area, the value programmed into the CDC register ranges 1 to 16 (1h to 10h).

### 4.3.11 Tally Counters

The SONIC-16 provides three 16-bit counters used for monitoring network statistics on the number of CRC errors, Frame Alignment errors, and missed packets. These registers rollover after the count of FFFFh is reached and produce an interrupt if enabled in the Interrupt Mask Register (IMR). These counters are unaffected by the RXEN bit in the CR, but are halted when the RST bit in the CR is set. The data written to these registers is inverted before being latched. This means that if a value of FFFFh is written to these registers by the system, they will contain and read back the value 0000h. Data is not inverted during a read operation. The Tally registers, therefore, are cleared by writing all "1's" to them. A software or hardware reset does not affect the tally counters.

**CRC Tally Counter Register (CRCT):** The CRCT is a 16-bit read/write register. This register is used to keep track of the number of packets received with CRC errors. After a packet is accepted by the address recognition logic, this register is incremented if a CRC error is detected. If the packet also contains a Frame Alignment error, this counter is not incremented.

**FAE Tally Counter Register (FAET):** The FAET is a 16-bit read/write register. This register is used to keep track of the number of packets received with frame alignment errors. After a packet is accepted by the address recognition logic, this register is incremented if a FAE error is detected.

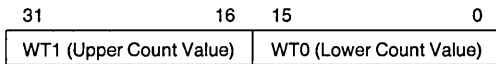
**Missed Packet Tally Counter Register (MPT):** The MPT is a 16-bit read/write register. After a packet is received, this counter is incremented if there is: (1) lack of memory resources to buffer the packet, (2) a FIFO overrun, or (3) a valid packet has been received, but the receiver is disabled (RXDIS is set in the command register).

## 4.0 SONIC-16 Registers (Continued)

### 4.3.12 General Purpose Timer

The SONIC-16 contains a 32-bit general-purpose watchdog timer for timing user-definable events. This timer is accessed by the user through two 16-bit read/write registers (WT1 and WT0). The lower count value is programmed through the WT0 register and the upper count value is programmed through the WT1 register.

These two registers are concatenated together to form the complete 32-bit timer. This timer, clocked at 1/2 the Transmit Clock (TXC) frequency, counts down from its programmed value and generates an interrupt, if enabled (Interrupt Mask register), when it rolls over from 0000 0000h to FFFF FFFFh. When the counter rolls over it continues decrementing unless explicitly stopped (setting the STP bit). The timer is controlled by the ST (Start Timer) and STP (Stop Timer) bits in the Command register. A hardware or software reset halts, but does not clear, the General Purpose timer.



### 4.3.13 Silicon Revision Register

This is a 16-bit read only register. It contains information on the current revision of the SONIC-16. The initial silicon begins at 0000h and subsequent revision will be incremented by one.

## 5.0 Bus Interface

SONIC-16 features a high speed non-multiplexed 23-bit address and 16-bit data bus designed for a wide range of sys-

tem environments. SONIC-16 contains an on-chip DMA and supplies all the necessary signals for DMA operation. With 23 address lines SONIC-16 can access a full 4 M-word address space. To accommodate different memory speeds wait states can be added to the bus cycle by two methods. The memory subsystem can add wait states by simply withholding the appropriate handshake signals. In addition, the SONIC-16 can be programmed (via the Data Configuration Register) to add wait states.

The SONIC-16 is designed to interface to both the National/Intel and Motorola style buses. To facilitate minimum chip count designs and complete bus compatibility the user can program the SONIC-16 for the following bus modes:

- National/Intel bus operating in synchronous mode
- National/Intel bus operating in asynchronous mode
- Motorola bus operating in synchronous mode
- Motorola bus operating in asynchronous mode

The mode pin (BMODE) along with the SBUS bit in the Data Configuration Register are used to select the bus mode.

This section describes the SONIC-16's pin signals, provides system interface examples, and describes the various SONIC-16 bus operations.

### 5.1 PIN CONFIGURATIONS

There are two user selectable pin configurations for SONIC-16 to provide the proper interface signals for either the National/Intel or Motorola style buses. The state of the BMODE pin is used to define the pin configuration. *Figure 5-1* shows the pin configuration when BMODE = 1 (tied to V<sub>CC</sub>) for the Motorola style bus. *Figure 5-2* shows the pin configuration when BMODE = 0 (tied to ground) for the National/Intel style bus.

5.0 Bus Interface (Continued)

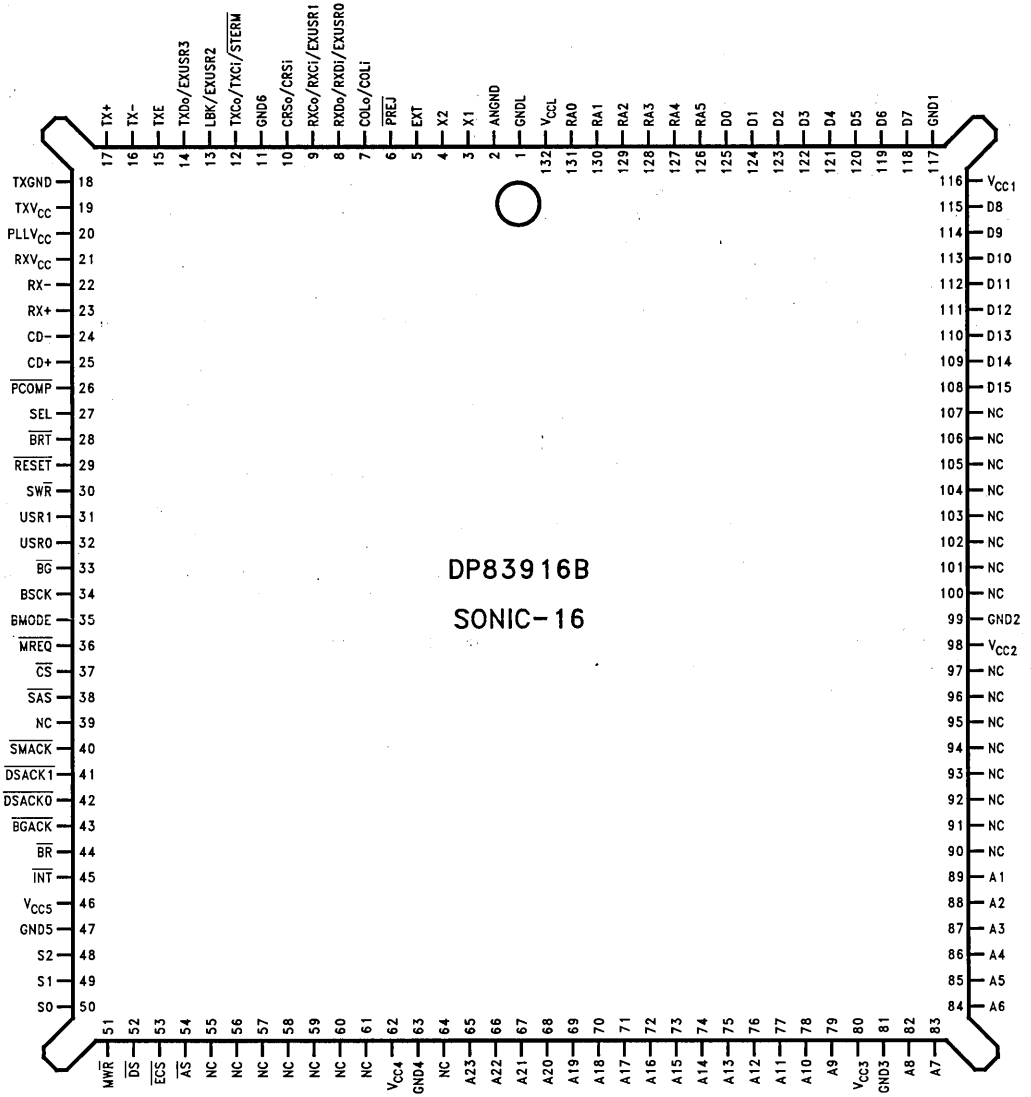


FIGURE 5-1. Connection Diagram (BMODE = 1)

TL/F/11722-23

5.0 Bus Interface (Continued)

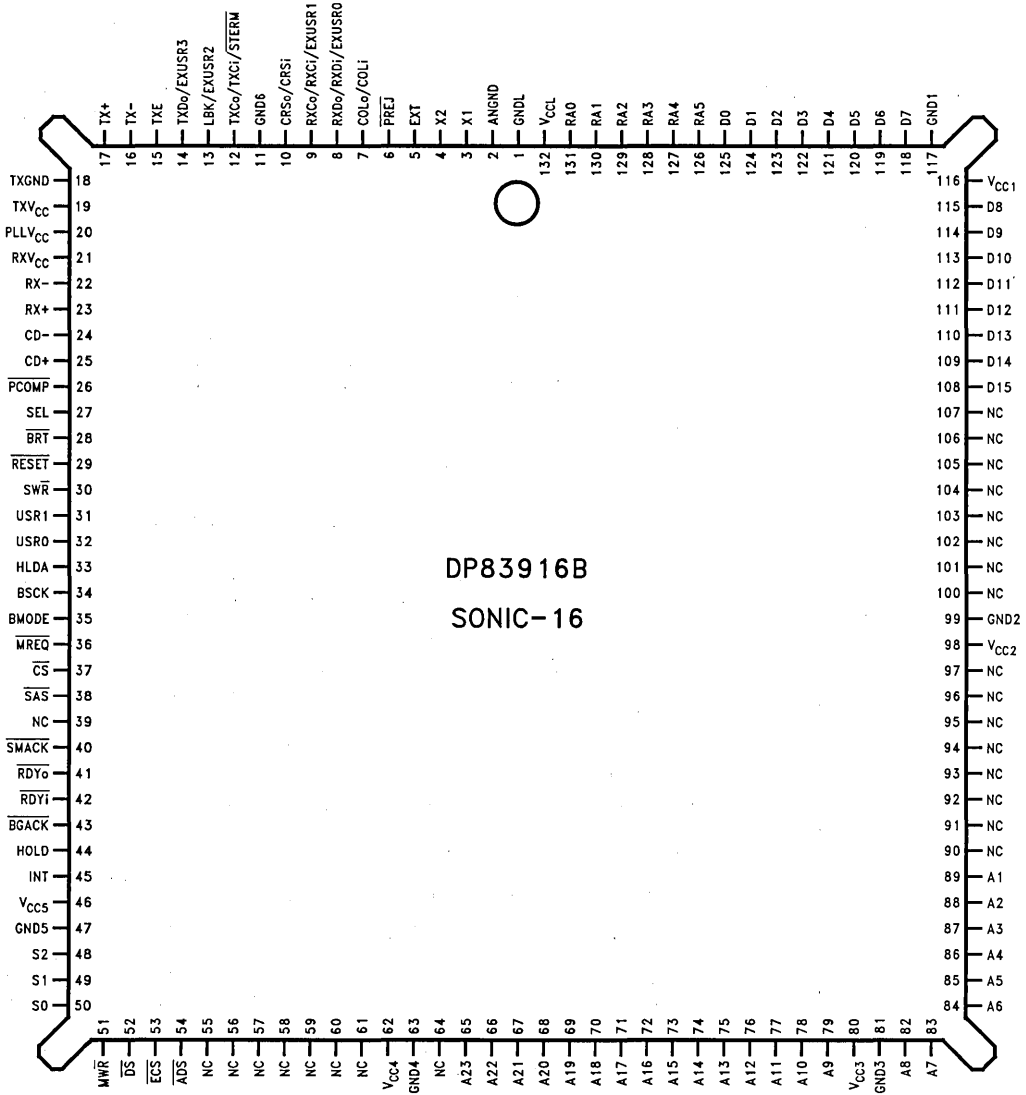


FIGURE 5-2. Connection Diagram (BMODE = 0)

TL/F/11722-24

## 5.0 Bus Interface (Continued)

### 5.2 PIN DESCRIPTION

I = input,  
 O = output, and  
 Z = TRI-STATE Inputs are TTL compatible  
 ECL = ECL-like drivers for interfacing to the AUI interface.  
 TP = Totem pole like drivers. These drivers are driven either high or low and are always driven. Drive levels are CMOS compatible.

TRI = TRI-STATE drivers. These pins are driven high, low or TRI-STATE. Drive levels are CMOS compatible. These pins may also be inputs (depending on the pin).

OC = Open Collector type drivers. These drivers are TRI-STATE when inactive and are driven low when active. These pins may also be inputs (depending on the pin).

TABLE 5-1. Pin Description

Symbol	Driver Type	Direction	Description
<b>NETWORK INTERFACE PINS</b>			
EXT		I	<b>External ENDEC Select:</b> Tying this pin to $V_{CC}$ (EXT = 1) disables the internal ENDEC and allows an external ENDEC to be used. Tying this pin to ground (EXT = 0) enables the internal ENDEC. This pin must be tied either to $V_{CC}$ or ground. Note the alternate pin definitions for CRS <sub>o</sub> /CRS <sub>i</sub> , COL <sub>o</sub> /COL <sub>i</sub> , RXD <sub>o</sub> /RXD <sub>i</sub> , RXC <sub>o</sub> /RXC <sub>i</sub> , and TXC <sub>o</sub> /TXC <sub>i</sub> . When EXT = 0 the first pin definition is used and when EXT = 1 the second pin definition is used.
CD+		I	<b>Collision +:</b> The positive differential collision input from the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
CD-		I	<b>Collision -:</b> The negative differential collision input from the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
RX+		I	<b>Receive +:</b> The positive differential receive data input from the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1)
RX-		I	<b>Receive -:</b> The negative differential receive data input from the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1)
TX+	ECL	O	<b>Transmit +:</b> The positive differential transmit output to the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
TX-	ECL	O	<b>Transmit -:</b> The negative differential transmit output to the transceiver. This pin should be unconnected when an external ENDEC is selected (EXT = 1).
CRS <sub>o</sub> CRS <sub>i</sub>	TP	O I	<b>Carrier Sense Output (CRS<sub>o</sub>)</b> from the internal ENDEC (EXT = 0): When EXT = 0 the CRS <sub>o</sub> signal is internally connected between the ENDEC and MAC units. It is asserted on the first valid high-to-low transition in the receive data (RX+ / -). This signal remains active 1.5 bit times after the last bit of data. Although this signal is used internally by the SONIC-16 it is also provided as an output to the user. <b>Carrier Sense Input (CRS<sub>i</sub>)</b> from an external ENDEC (EXT = 1): The CRS <sub>i</sub> signal is activated high when the external ENDEC detects valid data at its receive inputs.
COL <sub>o</sub> COL <sub>i</sub>	TP	O I	<b>Collision Output (COL<sub>o</sub>)</b> from the internal ENDEC (EXT = 0): When EXT = 0 the COL <sub>o</sub> signal is internally connected between the ENDEC and MAC units. This signal generates an active high signal when the 10 MHz collision signal from the transceiver is detected. Although this signal is used internally by the SONIC-16 it is also provided as an output to the user. <b>Collision Detect Input (COL<sub>i</sub>)</b> from an external ENDEC (EXT = 1): The COL <sub>i</sub> signal is activated from an external ENDEC when a collision is detected. This pin is monitored during transmissions from the beginning of the Start Of Frame Delimiter (SFD) to the end of the packet. At the end of transmission, this signal is monitored by the SONIC-16 for CD heartbeat.



## 5.0 Bus Interface (Continued)

TABLE 5-1. Pin Description (Continued)

Symbol	Driver Type	Direction	Description
<b>NETWORK INTERFACE PINS</b> (Continued)			
RXDo RXDi EXUSR0	TP  TRI	O I O, Z	<p>This pin will be TRI-STATE until the DCR has been written to. (See section 4.3.2, EXBUS, for more information.)</p> <p><b>Receive Data Output (RXDo)</b> from the internal ENDEC (EXT = 0): NRZ data output. When EXT = 0 the RXDOUT signal is internally connected between the ENDEC and MAC units. This signal must be sampled on the rising edge of the receive clock output (RXCo). Although this signal is used internally by the SONIC-16 it is also provided as an output to the user.</p> <p><b>Receive Data Input (RXDi)</b> from an external ENDEC (EXT = 1): The NRZ data decoded from the external ENDEC. This data is clocked in on the rising edge of RXCi.</p> <p><b>Extended User Output (EXUSR0):</b> When EXBUS has been set (see section 4.3.2), this pin becomes a programmable output. It will remain TRI-STATE until the SONIC-16 becomes a bus master, at which time it will be driven according to the value programmed in the DCR2 (Section 4.3.7).</p>
RXCo RXCi EXUSR1	TP  TRI	O I O, Z	<p>This pin will be TRI-STATE until the DCR has been written to. (See section 4.3.2, EXBUS, for more information.)</p> <p><b>Receive Clock Output (RXCo)</b> from the internal ENDEC (EXT = 0): When EXT = 0 the RXCo signal is internally connected between the ENDEC and MAC units. This signal is the separated receive clock from the Manchester data stream. It remains active 5-bit times after the deassertion of CRS0. Although this signal is used internally by the SONIC-16 it is also provided as an output to the user.</p> <p><b>Receive Clock Input (RXCi)</b> from an external ENDEC (EXT = 1): The separated received clock from the Manchester data stream. This signal is generated from an external ENDEC.</p> <p><b>Extended User Output (EXUSR1):</b> When EXBUS has been set (see section 4.3.2), this pin becomes a programmable output. It will remain TRI-STATE until the SONIC-16 becomes a bus master, at which time it will be driven according to the value programmed in the DCR2 (Section 4.3.7).</p>
TXD EXUSR3	TP TRI	O O, Z	<p>This pin will be TRI-STATE until the DCR has been written to. (See section 4.3.2, EXBUS, for more information.)</p> <p><b>Transmit Data (TXD):</b> The serial NRZ data from the MAC unit which is to be decoded by an external ENDEC. Data is valid on the rising edge of TXC. Although this signal is used internally by the SONIC-16 it is also provided as an output to the user.</p> <p><b>Extended User Output (EXUSR3):</b> When EXBUS has been set (see section 4.3.2), this pin becomes a programmable output. It will remain TRI-STATE until the SONIC-16 becomes a bus master, at which time it will be driven according to the value programmed in the DCR2 (Section 4.3.7).</p>
TXE	TP	O	<p><b>Transmit Enable:</b> This pin is driven high when the SONIC-16 begins transmission and remains active until the last byte is transmitted. Although this signal is used internally by the SONIC-16 it is also provided as an output to the user.</p>
TXCo TXCi STERM	TRI	O, Z I I	<p>This pin will be TRI-STATE until the DCR has been written to. (See section 4.3.2, EXBUS, for more information.)</p> <p><b>Transmit Clock Output (TXCo)</b> from the internal ENDEC (EXT = 0): This 10 MHz clock transmit clock output is derived from the 20 MHz oscillator. When EXT = 0 the TXCOUT signal is internally connected between the ENDEC and MAC units. Although this signal is used internally by the SONIC-16 it is also provided as an output to the user.</p> <p><b>Transmit Clock Input (TXCi)</b> (EXT = 1): This input clock from an external ENDEC is used for shifting data out of the MAC unit serializer. This clock is nominally 10 MHz.</p> <p><b>Synchronous Termination (STERM):</b> When the SONIC-16 is a bus master, it samples this pin before terminating its memory cycle. This pin is sampled synchronously and may only be used in asynchronous bus mode when BMODE = 1. See section 5.4.5 for more details.</p>

## 5.0 Bus Interface (Continued)

TABLE 5-1. Pin Description (Continued)

Symbol	Driver Type	Direction	Description
<b>NETWORK INTERFACE PINS</b> (Continued)			
LBK EXUSR2	TP TRI	O O, Z	<p>This pin will be TRI-STATE until the DCR has been written to. (See Section 4.3.2, EXBUS, for more information.)</p> <p><b>Loopback (LBK):</b> When ENDEC loopback is programmed, this pin is asserted high. Although this signal is used internally by the SONIC-16 it is also provided as an output to the user.</p> <p><b>Extended User Output (EXUSR2):</b> When EXBUS has been set (see Section 4.3.2), this pin becomes a programmable output. It will remain TRI-STATE until the SONIC-16 becomes a bus master, at which time it will be driven according to the value programmed in the DCR2 (Section 4.3.7).</p>
PCOMP	TRI	O, Z	<p><b>Packet Compression:</b> This pin is used with the Management Bus of the DP83950, Repeater Interface Controller (RIC). The SONIC-16 can be programmed to assert PCOMP whenever there is a CAM match, or when there is not a match. The RIC uses this signal to compress (shorten) a received packet for management purposes and to reduce memory usage. (See the DP83950 datasheet for more details on the RIC Management Bus.) The operation of this pin is controlled by bits 1 and 2 in the DCR2 register. PCOMP will remain TRI-STATE until these bits are written to.</p>
SEL		I	<p><b>Mode Select (EXT = 0):</b> This pin is used to determine the voltage relationship between TX+ and TX- during idle at the primary of the isolation transformer on the network interface. When tied to V<sub>CC</sub>, TX+ and TX- are at equal voltages during idle. When tied to ground, the voltage at TX+ is positive with respect to TX- during idle on the primary side of the isolation transformer (Figure 6-2).</p>
PREJ		I, O	<p><b>Packet Reject:</b> This signal is used to reject received packets. When asserted low for at least two receive clocks (RXC), the SONIC-16 will reject the incoming packet. This pin can be asserted up to the 2nd to the last bit of reception to reject a packet.</p>
X1	TP	I	<p><b>Crystal or External Oscillator Input:</b> This signal is used to provide clocking signals for the internal ENDEC. A crystal can be connected to this pin along with X2, or an oscillator module may be used. Typically the output of an oscillator module is connected to this pin. See Section 6.1.3 for more information about using oscillators or crystals.</p>
X2		I, O	<p><b>Crystal Feedback Output:</b> This signal is used to provide clocking signals for the internal ENDEC. A crystal may be connected to this pin along with X1, or an oscillator module may be used. See Section 6.1.3 for more information about using oscillator modules or crystals.</p>
<b>BUS INTERFACE PINS</b>			
BMODE		I	<p><b>Bus Mode:</b> This input enables the SONIC-16 to be compatible with standard microprocessor buses. The level of this pin affects byte ordering (little or big endian) and controls the operation of the bus interface control signals. A high level (tied to V<sub>CC</sub>) selects Motorola mode (big endian) and a low level (tied to ground) selects National/Intel mode (little endian). Note the alternate pin definitions for AS/ADS, MRW/MWR, INT/INT, BR/HOLD, BG/HLDA, SRW/SWR, DSACK0/RDYi, and DSACK1/RDYo. When BMODE = 1 the first pin definition is used and when BMODE = 0 the second pin definition is used. See Sections 5.4.1, 5.4.4, and 5.4.5.</p>
D31-D0	TRI	I, O, Z	<p><b>Data Bus:</b> These bidirectional lines are used to transfer data on the system bus. When the SONIC-16 is a bus master, 16-bit data is transferred on D15-D0 and 32-bit data is transferred on D31-D0. When the SONIC-16 is accessed as a slave, register data is driven onto lines D15-D0.</p>

## 5.0 Bus Interface (Continued)

TABLE 5-1. Pin Description (Continued)

Symbol	Driver Type	Direction	Description
<b>BUS INTERFACE PINS (Continued)</b>			
A31–A1	TRI	O, Z	<b>Address Bus:</b> These signals are used by the SONIC-16 to drive the DMA address after the SONIC-16 has acquired the bus. Since the SONIC-16 aligns data to word boundaries, only 23 address lines are needed.
RA5–RA0		I	<b>Register Address Bus:</b> These signals are used to access SONIC-16's internal registers. When the SONIC-16 is accessed, the CPU drives these lines to select the desired SONIC-16 register.
$\overline{AS}$ ADS	TRI TRI	I, O, Z O, Z	<b>Address Strobe (<math>\overline{AS}</math>):</b> When BMODE = 1, the falling edge indicates valid status and address. The rising edge indicates the termination of the memory cycle. <b>Address Strobe (ADS):</b> When BMODE = 0, the rising edge indicates valid status and address.
MRW MWR	TRI TRI	O, Z O, Z	When the SONIC-16 has acquired the bus, this signal indicates the direction of data. <b>Memory Read/Write Strobe (MRW):</b> When BMODE = 1, this signal is high during a read cycle and low during a write cycle. <b>Memory Read/Write Strobe (MWR):</b> When BMODE = 0, the signal is low during a read cycle and high during a write cycle.
$\overline{INT}$ INT	OC TP	O, Z O	Indicates that an interrupt (if enabled) is pending from one of the sources indicated by the Interrupt Status register. Interrupts that are disabled in the Interrupt Mask register will not activate this signal. <b>Interrupt (<math>\overline{INT}</math>):</b> This signal is active low when BMODE = 1. <b>Interrupt (INT):</b> This signal is active high when BMODE = 0.
RESET		I	<b>Reset:</b> This signal is used to hardware reset the SONIC-16. When asserted low, the SONIC-16 transitions into the reset state after 10 transmit clocks or 10 bus clocks if the bus clock period is greater than the transmit clock period.
S2–S0	TP	O	<b>Bus Status:</b> These three signals provide a continuous status of the current SONIC-16 bus operations. See Section 5.4.3 for status definitions.
BCLK		I	<b>Bus Clock:</b> This clock provides the timing for the SONIC-16 DMA engine.
$\overline{BR}$ HOLD	OC TP	O, Z O	<b>Bus Request (<math>\overline{BR}</math>):</b> When BMODE = 1, the SONIC-16 asserts this pin low when it attempts to gain access to the bus. When inactive this signal is at TRI-STATE. <b>Hold Request (HOLD):</b> When BMODE = 0, the SONIC-16 drives this pin high when it intends to use the bus and is driven low when inactive.
$\overline{BG}$ HLDA		I I	<b>Bus Grant (<math>\overline{BG}</math>):</b> When BMODE = 1 this signal is a bus grant. The system asserts this pin low to indicate potential mastership of the bus. <b>Hold Acknowledge (HLDA):</b> When BMODE = 0 this signal is used to inform the SONIC-16 that it has attained the bus. When the system asserts this pin high, the SONIC-16 has gained ownership of the bus.
$\overline{BGACK}$	TRI	I, O, Z	<b>Bus Grant Acknowledge:</b> When BMODE = 1, the SONIC-16 asserts this pin low when it has determined that it can gain ownership of the bus. The SONIC-16 checks the following signal before driving $\overline{BGACK}$ . 1) $\overline{BG}$ has been received through the bus arbitration process. 2) $\overline{AS}$ is deasserted, indicating that the CPU has finished using the bus. 3) $\overline{DSACK0}$ and $\overline{DSACK1}$ are deasserted, indicating that the previous slave device is off the bus. 4) $\overline{BGACK}$ is deasserted, indicating that the previous master is off the bus. This pin is only used when BMODE = 1.

## 5.0 Bus Interface (Continued)

TABLE 5-1. Pin Description (Continued)

Symbol	Driver Type	Direction	Description
<b>BUS INTERFACE PINS (Continued)</b>			
$\overline{CS}$		I	<b>Chip Select:</b> The system asserts this pin low to access the SONIC-16's registers. The registers are selected by placing an address on lines RA5–RA0. <b>Note:</b> Both $\overline{CS}$ and $\overline{MREQ}$ must not be asserted concurrently. If these signals are successively asserted, there must be at least two bus clocks between the deasserting edge of the first signal and the asserting edge of the second signal.
$\overline{SAS}$		I	<b>Slave Address Strobe:</b> The system asserts this pin to latch the register address on lines RA0–RA5. When BMODE = 1, the address is latched on the falling edge of $\overline{SAS}$ . When BMODE = 0 the address is latched on the rising edge of $\overline{SAS}$ .
$\overline{SDS}$		I	<b>Slave Data Strobe:</b> The system asserts this pin to indicate valid data is on the bus during a register write operation or when data may be driven onto the bus during a register read operation.
$\overline{SRW}$ SWR		I I	The system asserts this pin to indicate whether it will read from or write to the SONIC-16's registers. <b>Slave Read/Write (<math>\overline{SRW}</math>):</b> When BMODE = 1, this signal is asserted high during a read and low during a write. <b>Slave Read/Write Strobe (<math>\overline{SWR}</math>):</b> when BMODE = 0, this signal is asserted low during a read and high during a write.
$\overline{DS}$	TRI	O, Z	<b>Data Strobe:</b> When the SONIC-16 is bus master, it drives this pin low during a read cycle to indicate that the slave device may drive data onto the bus; in a write cycle, this pin indicates that the SONIC-16 has placed valid data onto the bus.
$\overline{DSACK0}$ $\overline{RDYi}$ $\overline{DSACKT}$ $\overline{RDY0}$	TRI TRI TRI	I, O, Z I I, O, Z O, Z	<b>Data and Size Acknowledge 0 and 1 (<math>\overline{DSACK0,1}</math> BMODE = 1):</b> These pins are the output slave acknowledge to the system when the SONIC-16 registers have been accessed and the input slave acknowledgement when the SONIC-16 is busmaster. When a register has been accessed, the SONIC-16 drives the $\overline{DSACK0,1}$ pins low to terminate the slave cycle. (Note that the SONIC-16 responds as a 32-bit peripheral, but drives data only on lines D0–D15). When the SONIC-16 is bus master, it samples these pins before terminating its memory cycle. These pins are sampled synchronously or asynchronously depending on the state of the SBUS bit in the Data Configuration register. See Section 5.4.5 for details. Note that the SONIC-16 does not allow dynamic bus sizing. <b>Ready Input (<math>\overline{RDYi}</math>, BMODE = 0):</b> When the SONIC-16 is a bus master, the system asserts this signal high to insert wait-states and low to terminate the memory cycle. This signal is sampled synchronously or asynchronously depending on the state of the SBUS bit. See Sections 5.4.5 and 4.3.2 for details. <b>Ready Output (<math>\overline{RDY0}</math>, BMODE = 0):</b> When a register is accessed, the SONIC-16 asserts this signal to terminate the slave cycle.
BRT		I	<b>Bus Retry:</b> When the SONIC-16 is bus master, the system asserts this signal to rectify a potentially correctable bus error. This pin has 2 modes. Mode 1 (the LBR in the Data Configuration register is set to 0): Assertion of this pin forces the SONIC-16 to terminate the current bus cycle and will repeat the same cycle after BRT has been deasserted. Mode 2 (the LBR bit in the Data Configuration register is set to 1): Assertion of this signal forces the SONIC-16 to retry the bus operation as in Mode 1. However, the SONIC-16 will not continue DMA operations until the BR bit in the ISR is reset.
$\overline{ECS}$	TRI	O, Z	<b>Early Cycle Start:</b> This output gives the system earliest indication that a memory operation is occurring. This signal is driven low at the rising edge of T1 and high at the falling edge of T1.

## 5.0 Bus Interface (Continued)

TABLE 5-1. Pin Description (Continued)

Symbol	Driver Type	Direction	Description
<b>SHARED-MEMORY ACCESS PINS</b>			
MREQ		I	<b>Memory Request:</b> The system asserts this signal low when it attempts to access the shared-buffer RAM. The on-chip arbiter resolves accesses between the system and the SONIC-16. <b>Note:</b> Both $\overline{CS}$ and $\overline{MREQ}$ must not be asserted concurrently. If these signals are successively asserted, there must be at least two bus clocks between the deasserting edge of the first signal and the asserting edge of the second signal.
SMACK	TP	O	<b>Slave and Memory Acknowledge:</b> SONIC-16 asserts this dual function pin low in response to either a Chip Select ( $\overline{CS}$ ) or a Memory Request ( $\overline{MREQ}$ ) when the SONIC-16's registers or it's buffer memory is available for accessing. This pin can be used for enabling bus drivers for dual-bus systems.
<b>USER DEFINABLE PINS</b>			
USR0,1	TRI	I, O, Z	<b>User Define 0,1:</b> These signals are inputs when SONIC-16 is hardware reset and are outputs when SONIC-16 is a bus master ( $\overline{HLDA}$ or $\overline{BGACK}$ ). When hard reset ( $\overline{RST}$ ) is low, these signals input directly into bits 8 and 9 of the Data Configuration register (DCR) respectively. The levels on these pins are latched on the rising edge of $\overline{RST}$ . During busmaster operations ( $\overline{HLDA}$ or $\overline{BGACK}$ is active), these pins are outputs whose levels are programmable through bits 11 and 12 of the DCR respectively. The USR0,1 pins should be pulled up to $V_{CC}$ or pulled down to ground. A 4.7 k $\Omega$ pull-up resistor is recommended.
<b>POWER AND GROUND PINS</b>			
VCC1-5			<b>Power:</b> The +5V power supply for the digital portions of the SONIC-16.
TXVCC RXVCC PLLVCC VCCL			<b>Power:</b> These pins are the +5V power supply for the SONIC-16 ENDEC unit. These pins must be tied to $V_{CC}$ even if the internal ENDEC is not used.
GND1-6			<b>Ground:</b> The ground reference for the digital portions of the SONIC-16.
TXGND ANGND GNDL			<b>Ground:</b> These pins are the ground references for the SONIC-16 ENDEC unit. These pins must be tied to ground even if the internal ENDEC is not used.

## 5.0 Bus Interface (Continued)

### 5.3 SYSTEM CONFIGURATION

Any device that meets the SONIC-16 interface protocol and electrical requirements (timing, threshold, and loading) can be interfaced to SONIC-16. Since two bus protocols are provided, via the BMODE pin, the SONIC-16 can interface directly to most microprocessors. *Figure 5-3* shows a typical interface to the National/Intel style bus (BMODE=0) and *Figure 5-4* shows a typical interface to the Motorola style bus (BMODE=1).

The BMODE pin also controls byte ordering. When BMODE=1 big endian byte ordering is selected and when BMODE=0 little endian byte ordering is selected.

### 5.4 BUS OPERATIONS

There are two types of system bus operations: 1) SONIC-16 as a slave, and 2) SONIC-16 as a bus master. When SONIC-16 is a slave (e.g., a CPU accessing SONIC-16 registers) all transfers are non-DMA. When SONIC-16 is a bus master (e.g., SONIC-16 accessing receive or transmit buffer/descriptor areas) all transfers are block transfers using SONIC-16's on-chip DMA. This section describes the SONIC-16 bus operations. Pay special attention to all sections labeled as "Note". These conditions must be met for proper bus operation.

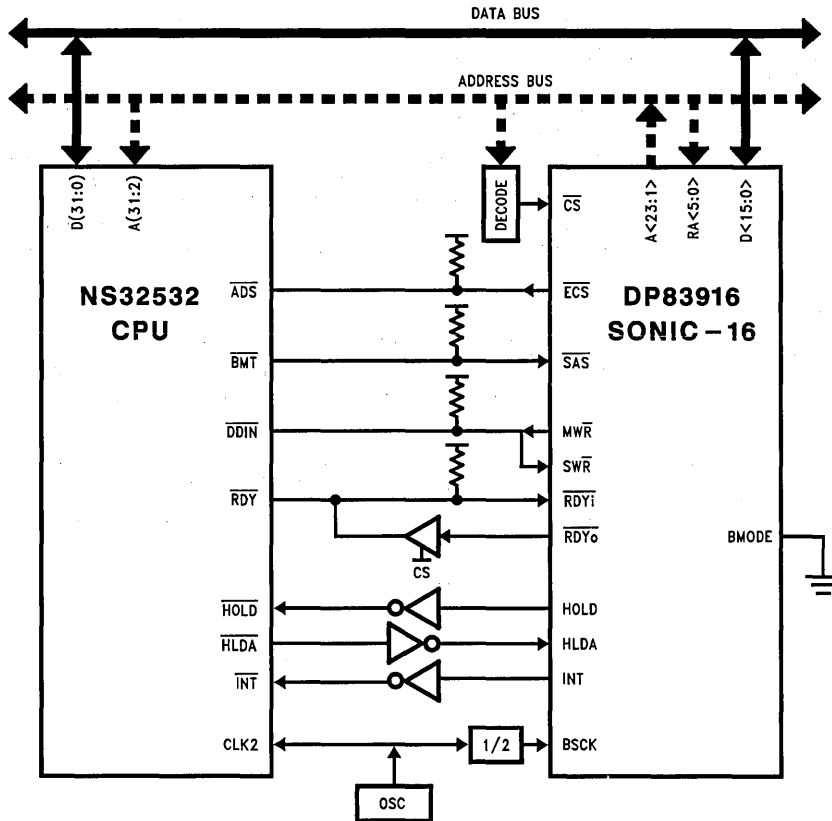


FIGURE 5-3. SONIC-16 to NS32532 Interface Example

TL/F/11722-25

5.0 Bus Interface (Continued)

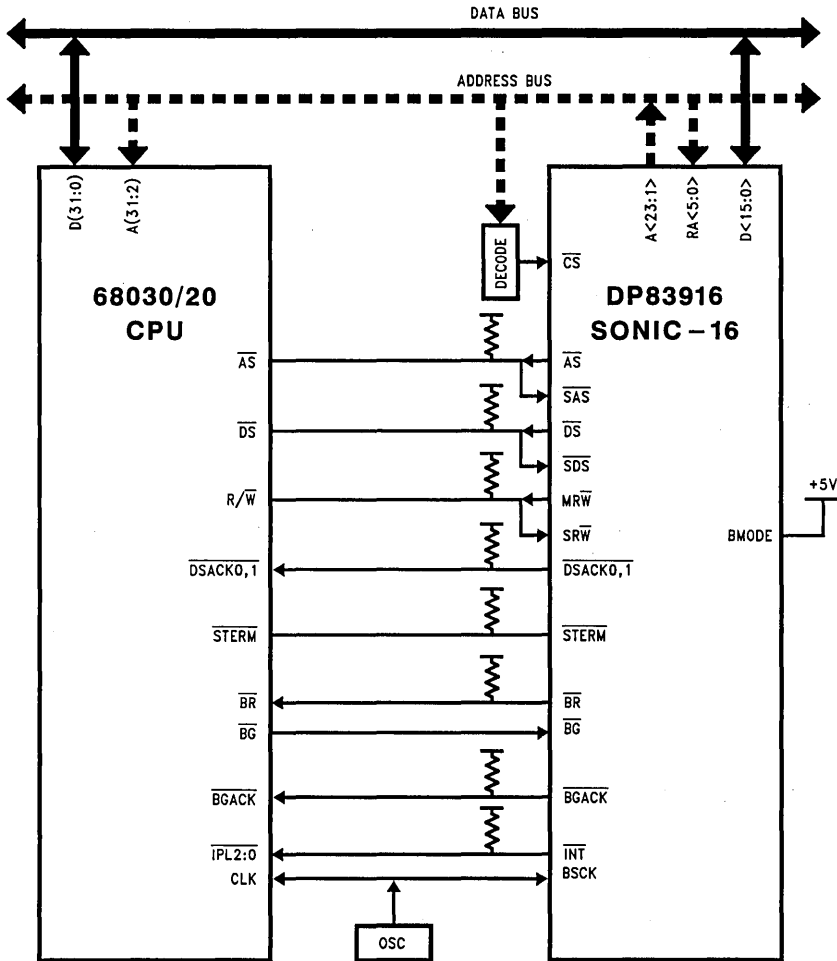


FIGURE 5-4. SONIC-16 to Motorola 68030/20 Interface Example

TL/F/11722-26

## 5.0 Bus Interface (Continued)

### 5.4.1 Acquiring The Bus

The SONIC-16 requests the bus when 1) its FIFO threshold has been reached or 2) when the descriptor areas in memory (i.e., RRA, RDA, CDA, and TDA) are accessed. Note that when the SONIC-16 moves from one area in memory to another (e.g., RBA to RDA), it always deasserts its bus request and then requests the bus again when accessing the next area in memory.

The SONIC-16 provides two methods to acquire the bus for compatibility with National/Intel or Motorola type microprocessors. These two methods are selected by setting the proper level on the BMODE pin.

Figures 5-5 and 5-6 show the National/Intel (BMODE = 0) and Motorola (BMODE = 1) bus request timing. Descriptions of each mode follows. For both modes, when the SONIC-16 relinquishes the bus, there is an extra holding state (Th) for one bus cycle after the last DMA cycle (T2). This assures that the SONIC-16 does not contend with another bus master after it has released the bus.

#### BMODE = 0

The National/Intel processors require a 2-way handshake using a HOLD REQUEST/HOLD ACKNOWLEDGE protocol (Figure 5-5). When the SONIC-16 needs to access the bus, it issues a HOLD REQUEST (HOLD) to the microprocessor. The microprocessor, responds with a HOLD ACKNOWLEDGE (HLDA) to the SONIC-16. The SONIC-16 then begins its memory transfers on the bus. As long as the CPU maintains HLDA active, the SONIC-16 continues until it has finished its memory block transfer. The CPU, however, can preempt the SONIC-16 from finishing the block transfer by deasserting HLDA before the SONIC-16 deasserts HOLD. This allows a higher priority device to preempt the SONIC-16 from continuing to use the bus. The SONIC-16 will request the bus again later to complete any operation that it was doing at the time of preemption.

As shown in Figure 5-5, the SONIC-16 will assert HOLD to either the falling or rising edge of the bus clock (BSCK). The default is for HOLD to be asserted on the falling edge. Setting the PH bit in the DCR2 (see Section 4.3.7) causes HOLD to be asserted  $\frac{1}{2}$  bus clock later on the rising edge (shown by the dotted line). Before HOLD is asserted, the SONIC-16 checks the HLDA line. If HLDA is asserted, HOLD will not be asserted until after HLDA has been deasserted first.

#### BMODE = 1

The Motorola protocol requires a 3-way handshake using a BUS REQUEST, BUS GRANT, and BUS GRANT ACKNOWLEDGE handshake (Figure 5-6). When using this protocol, the SONIC-16 requests the bus by lowering BUS REQUEST ( $\overline{BR}$ ). The CPU responds by issuing BUS GRANT ( $\overline{BG}$ ). Upon receiving  $\overline{BG}$ , the SONIC-16 assures that all devices have relinquished control of the bus before using the bus. The following signals must be deasserted before the SONIC-16 acquires the bus:

$\overline{BGACK}$   
 $\overline{AS}$   
 $\overline{DSACK0,T}$   
 $\overline{STERM}$  (Asynchronous Mode Only)

Deasserting  $\overline{BGACK}$  indicates that the previous master has released the bus. Deasserting  $\overline{AS}$  indicates that the previous master has completed its cycle and deasserting  $\overline{DSACK0,T}$  and  $\overline{STERM}$  indicates that the previous slave has terminated its connection to the previous master. The SONIC-16 maintains its mastership of the bus until it deasserts  $\overline{BGACK}$ . It can not be preempted from the bus.

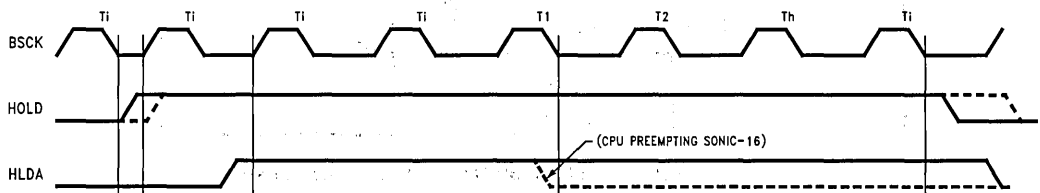


FIGURE 5-5. Bus Request Timing, BMODE = 0

TL/F/11722-27



## 5.0 Bus Interface (Continued)

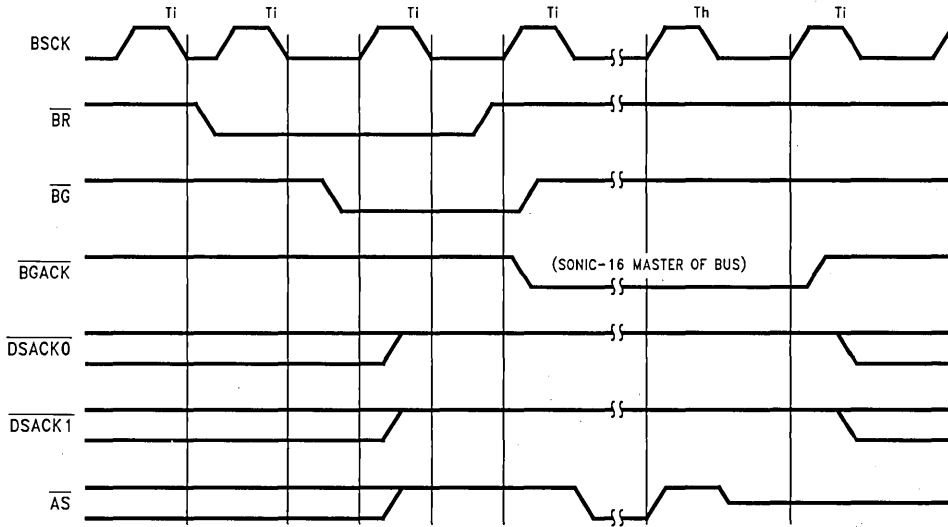


FIGURE 5-6. Bus Request Timing, BMODE = 1

TL/F/11722-28

### 5.4.2 Block Transfers

The SONIC-16 performs block operations during all bus actions, thereby providing efficient transfers to memory. The block cycle consists of three parts. The first part is the bus acquisition phase, as discussed above, in which the SONIC-16 gains access to the bus. Once it has access of the bus, the SONIC-16 enters the second phase by transferring data to/from its internal FIFOs or registers from/to memory. The SONIC-16 transfers data from its FIFOs in either EXACT BLOCK mode or EMPTY/FILL.

**EXACT BLOCK mode:** In this mode the number of words (or long words) transferred during a block transfer is determined by either the Transmit or Receive FIFO thresholds programmed in the Data Configuration Register.

**EMPTY/FILL mode:** In this mode the DMA completely fills the Transmit FIFO during transmission, or completely empties the Receive FIFO during reception. This allows for greater bus latency.

When the SONIC-16 accesses the Descriptor Areas (i.e., RRA, RDA, CDA, and TDA), it transfers data between its registers and memory. All fields which need to be used are accessed in one block operation. Thus, the SONIC-16 performs 4 accesses in the RRA (see Section 3.4.4.2), 7 accesses in the RDA (see Section 3.4.6.1), 2, 3, or 6 accesses in the TDA (see Section 3.5.4) and 4 accesses in the CDA.

### 5.4.3 Bus Status

The SONIC-16 presents three bits of status information on pins S2-S0 which indicate the type of bus operation the SONIC-16 is currently performing (Table 5-2). Bus status is valid when at the falling edge of AS or the rising edge of ADS.

TABLE 5-2. Bus Status

S2	S1	S0	Status
1	1	1	The bus is idle. The SONIC-16 is not performing any transfers on the bus.
1	0	1	The Transmit Descriptor Area (TDA) is currently being accessed.
0	0	1	The Transmit Buffer Area (TBA) is currently being read.
0	1	1	The Receive Buffer Area (RBA) is currently being written to. Only data is being written, though, not a Source or Destination address.
0	1	0	The Receive Buffer Area (RBA) is currently being written to. Only the Source or Destination address is being written, though.
1	1	0	The Receive Resource Area (RRA) is currently being read.
1	0	0	The Receive Descriptor Area (RDA) is currently being accessed.
0	0	0	The CAM Descriptor Area (CDA) is currently being accessed.

## 5.0 Bus Interface (Continued)

### 5.4.3.1 Bus Status Transitions

When the SONIC-16 acquires the bus, it only transfers data to/from a single area in memory (i.e., TDA, TBA, RDA, RBA, RRA, or CDA). Thus, the bus status pins remain stable for the duration of the block transfer cycle with the following three exceptions: 1) If the SONIC-16 is accessed during a block transfer, S2-S0 indicates bus idle during the register access, then returns to the previous status. 2) If the SONIC-16 finishes writing the Source Address during a block transfer S2-S0 changes from [0,1,0] to [0,1,1]. 3) During an RDA access between the RXpkt.seq\_no and RXpkt.link access, and between the RXpkt.link and RXpkt.in\_use access, S2-S0 will respectively indicate idle [1,1,1] for 2 or 1 bus clocks. Status will be valid on the falling edge of  $\overline{AS}$  or rising edge of ADS.

Figure 5-7 illustrates the SONIC-16's transitions through memory during the process of transmission and reception. During transmission, the SONIC-16 reads the descriptor information from the TDA and then transmits data of the packet from the TBA. The SONIC-16 moves back and forth between the TDA and TBA until all fragments and packets are transmitted. During reception, the SONIC-16 takes one of two paths. In the first case (path A), when the SONIC-16 detects EOL=0 from the previous reception, it buffers the accepted packet into the RBA, and then writes the descriptor information to the RDA. If the RBA becomes depleted (i.e., RBWC0,1 < EOBC), it moves to the RRA to read a resource descriptor. In the second case (path B), when the SONIC-16 detects EOL=1 from the previous reception, it

rereads the RXpkt.link field to determine if the system has reset the EOL bit since the last reception. If it has, the SONIC-16 buffers the packet as in the first case. Otherwise, it rejects the packet and returns to idle.

### 5.4.4 Bus Mode Compatibility

For compatibility with different microprocessor and bus architectures, the SONIC-16 operates in one of two modes (set by the BMODE pin) called the National/Intel or little endian mode (BMODE tied low) and the Motorola or big endian mode (BMODE tied high). The definitions for several pins change depending on the mode the SONIC-16 is in. Table 5-3 shows these changes. These modes affect both master and slave bus operations with the SONIC-16.

TABLE 5-3. Bus Mode Compatibility

Pin Name	BMODE = 0 (National/Intel)	BMODE = 1 (Motorola)
$\overline{BR}$ /HOLD	HOLD	$\overline{BR}$
$\overline{BG}$ /HLDA	HLDA	$\overline{BG}$
MR $\overline{W}$ /MWR	MWR	MR $\overline{W}$
SR $\overline{W}$ /SWR	SWR	SR $\overline{W}$
$\overline{DSACK0}$ /RDYi	RDYi	$\overline{DSACK0}$
$\overline{DSACK1}$ /RDYo	RDYo	$\overline{DSACK1}$
$\overline{AS}$ /ADS	ADS	$\overline{AS}$
INT/INT	INT	INT

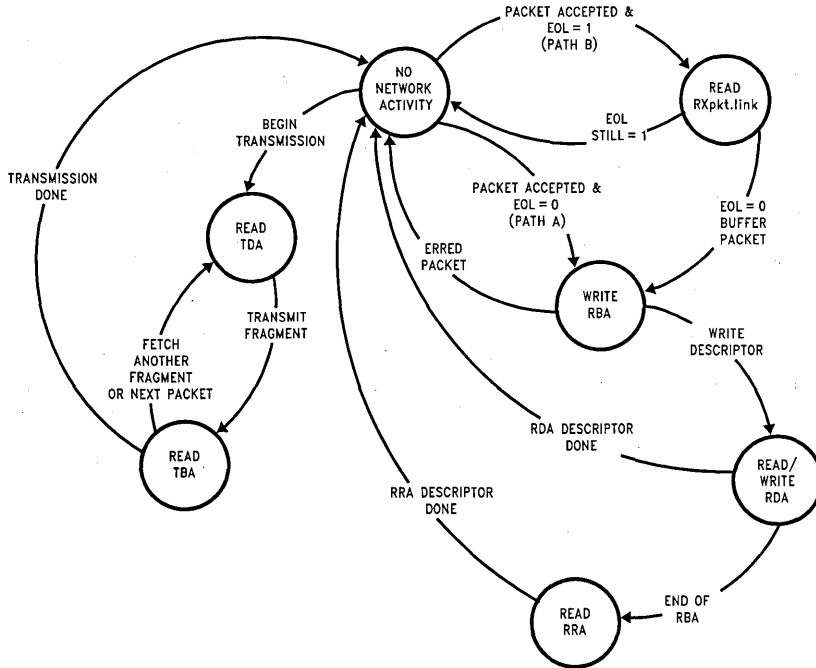


FIGURE 5-7. Bus Status Transitions

TL/F/11722-29

## 5.0 Bus Interface (Continued)

### 5.4.5 Master Mode Bus Cycles

In order to add additional compatibility with different bus architectures, there are two other modes that affect the operation of the bus. These modes are called the synchronous and asynchronous modes and are programmed by setting or resetting the SBUS bit in the Data Configuration Register (DCR). The synchronous and asynchronous modes do not have an effect on slave accesses to the SONIC-16 but they do affect the master mode operation. Within the particular bus/processor mode, synchronous and asynchronous modes are very similar. This section discusses all four modes of operation of the SONIC-16 (National/Intel vs. Motorola, synchronous vs. asynchronous) when it is a bus master.

In this section, the rising edge of T1 and T2 means the beginning of these states, and the falling edge of T1 and T2 means the middle of these states.

#### 5.4.5.1 Adding Wait States

To accommodate different memory speeds, the SONIC-16 provides two methods for adding wait states for its bus operations. Both of these methods can be used singly or in

conjunction with each other. A memory cycle is extended by adding additional T2 states. The first method inserts wait-states by withholding the assertion of  $\overline{DSACK0,1}$ / $\overline{STERM}$  or  $\overline{RDYi}$ . The other method allows software to program wait-states. Programming the WC0, WC1 bits in the Data Configuration Register allows 1 to 3 wait-states to be added on each memory cycle. These wait states are inserted between the T1 and T2 bus states and are called T2(wait) bus states. The SONIC-16 will not look at the  $\overline{DSACK0,1}$ ,  $\overline{STERM}$  or  $\overline{RDYi}$  lines until the programmed wait states have passed. Hence, in order to complete a bus operation that includes programmed wait states, the  $\overline{DSACK0,1}$ ,  $\overline{STERM}$  or  $\overline{RDYi}$  lines must be asserted at their proper times at the end of the cycle during the last T2, not during a programmed wait state. The only exception to this is asynchronous mode where  $\overline{DSACK0,1}$  or  $\overline{RDYi}$  would be asserted during the last programmed wait state, T2 (wait). See the timing for these signals in the timing diagrams for more specific information. Programmed wait states do not affect Slave Mode bus cycles.

## 5.0 Bus Interface (Continued)

### 5.4.5.2 Memory Cycle for BMODE = 1, Synchronous Mode

On the rising edge of T1, the SONIC-16 asserts  $\overline{ECS}$  to indicate that the memory cycle is starting. The address (A31-A1), bus status (S2-S0) and the direction strobe (MRW) are driven and do not change for the remainder of the memory cycle. On the falling edge of T1, the SONIC-16 deasserts  $\overline{ECS}$  and asserts  $\overline{AS}$ .

In synchronous mode,  $\overline{DSACK0,1}$  are sampled on the rising edge of T2. T2 states will be repeated until  $\overline{DSACK0,1}$  are

sampled properly in a low state.  $\overline{DSACK0,1}$  must meet the setup and hold times with respect to the rising edge of bus clock for proper operation.

During read cycles (Figure 5-8) data (D15-D0) is latched at the falling edge of T2 and  $\overline{DS}$  is asserted at the falling edge of T1. For write cycles (Figure 5-9) data is driven on the falling edge of T1. If there are wait states inserted,  $\overline{DS}$  is asserted on the falling edge of T2. The SONIC-16 terminates the memory cycle by deasserting  $\overline{AS}$  and  $\overline{DS}$  at the falling edge of T2.

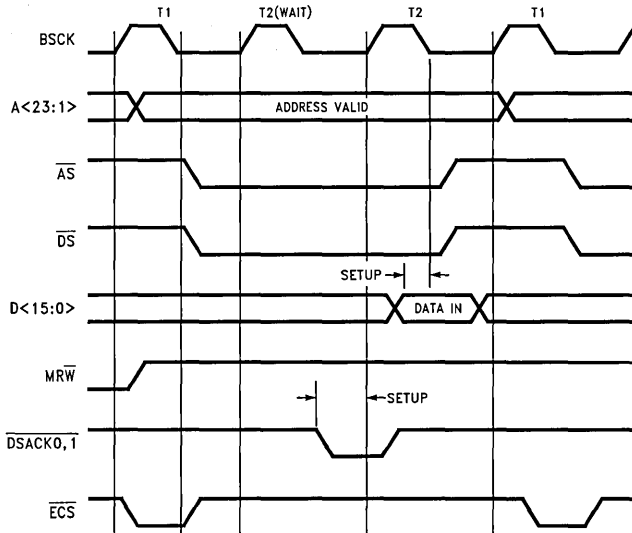


FIGURE 5-8. Memory Read, BMODE = 1, Synchronous (1 Wait-State)

TL/F/11722-31

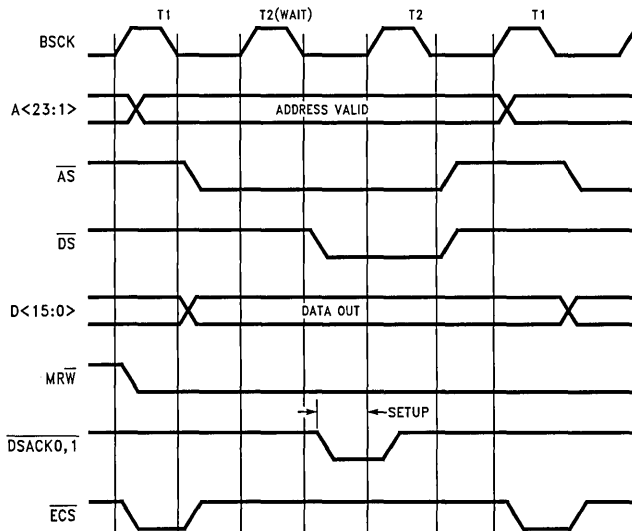


FIGURE 5-9. Memory Write, BMODE = 1, Synchronous (1 Wait-State)

TL/F/11722-33

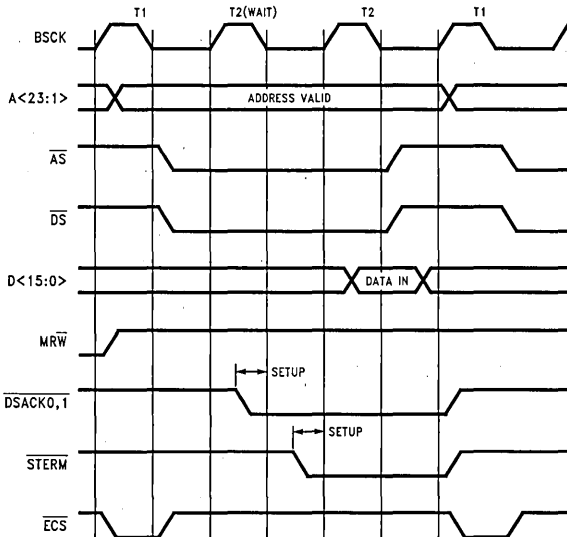
## 5.0 Bus Interface (Continued)

### 5.4.5.3 Memory Cycle for BMODE = 1, Asynchronous Mode

On the rising edge of T1, the SONIC-16 asserts  $\overline{ECS}$  to indicate that the memory cycle is starting. The address (A23-A1), bus status (S2-S0) and the direction strobe (MRW) are driven and do not change for the remainder of the memory cycle. On the falling edge of T1, the SONIC-16 deasserts  $\overline{ECS}$  and asserts  $\overline{AS}$ .

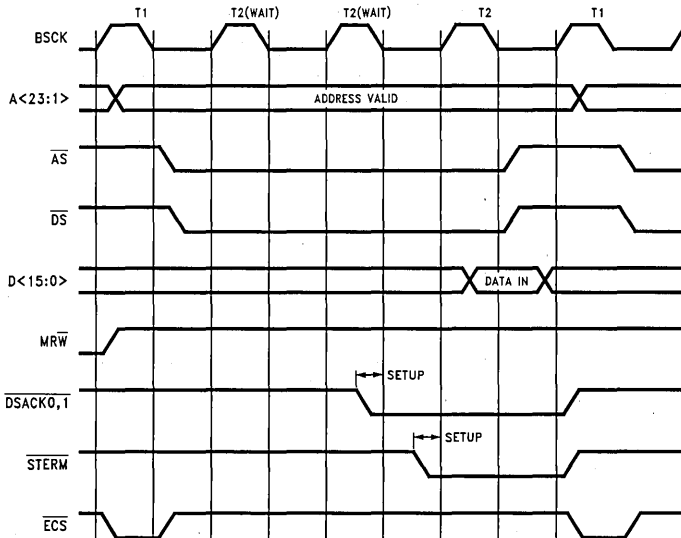
In asynchronous mode,  $\overline{DSACK0,1}$  are asynchronously sampled on the falling edge of both T1 and T2.  $\overline{DSACK0,1}$

do not need to be synchronized to the bus clock because the chip always resolves these signals to either a high or low state. If a synchronous termination of the bus cycle is required, however,  $\overline{STERM}$  may be used.  $\overline{STERM}$  is sampled on the rising edge of T2 and must meet the setup and hold times with respect to that edge for proper operation. Meeting the setup time for  $\overline{DSACK0,1}$  or  $\overline{STERM}$  guarantees that the SONIC-16 will terminate the memory cycle 1½



TL/F/11722-36

FIGURE 5-10. Memory Read, BMODE = 1, Asynchronous (1 Wait-State)



TL/F/11722-37

FIGURE 5-11. Memory Read, BMODE = 1, Asynchronous (2 Wait-State)

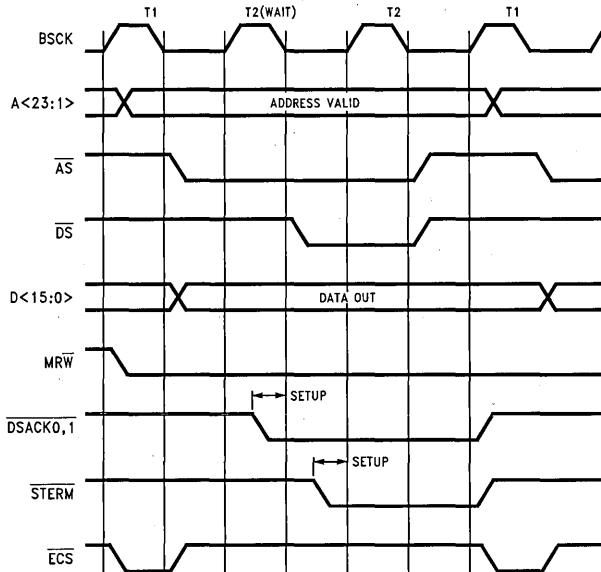
## 5.0 Bus Interface (Continued)

bus clocks after  $\overline{DSACK0,1}$  were sampled, or 1 cycle after  $\overline{STERM}$  was sampled. T2 states will be repeated until  $\overline{DSACK0,1}$  or  $\overline{STERM}$  are sampled properly in a low state. (see note below).

During read cycles (Figure 5-10 and 5-11), data (D15-D0) is latched at the falling edge of T2 and  $\overline{DS}$  is asserted at the falling edge of T1. For write cycles (Figures 5-12 and 5-13) data is driven on the falling edge of T1. If there are wait

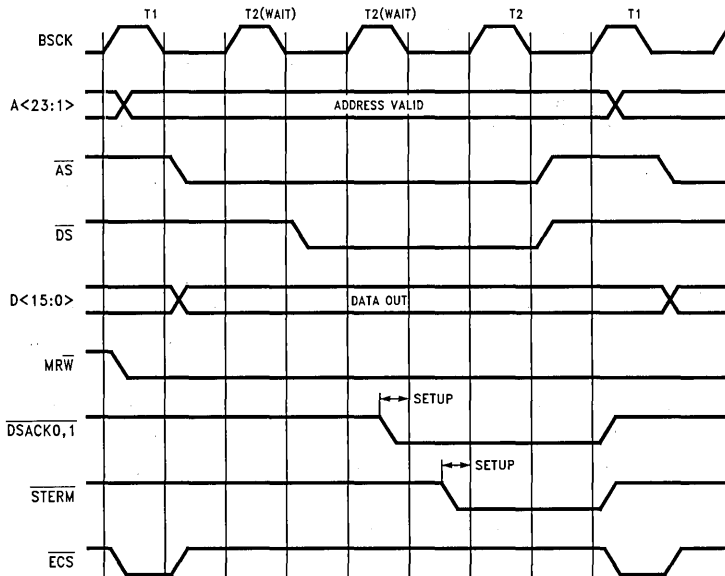
states inserted,  $\overline{DS}$  is asserted on the falling edge of the first T2(wait).  $\overline{DS}$  is not asserted for zero wait state write cycles. The SONIC-16 terminates the memory cycle by deasserting  $\overline{AS}$  and  $\overline{DS}$  at the falling edge of T2.

**Note:** If the setup time for  $\overline{DSACK0,1}$  is met during T1, or the setup time for  $\overline{STERM}$  is met during the first T2, the full asynchronous bus cycle will take only 2 bus clocks. This may be an unwanted situation. If so,  $\overline{DSACK0,1}$  and  $\overline{STERM}$  should be deasserted during T1 and the start of T2 respectively.



TL/F/11722-34

FIGURE 5-12. Memory Write, BMODE = 1, Asynchronous (1 Wait-State)



TL/F/11722-35

FIGURE 5-13. Memory Write, BMODE = 1, Asynchronous (2 Wait-State)

## 5.0 Bus Interface (Continued)

### 5.4.5.4 Memory Cycle for BMODE = 0, Synchronous Mode

On the rising edge of T1, the SONIC-16 asserts  $\overline{ADS}$  and  $\overline{ECS}$  to indicate that the memory cycle is starting. The address (A23-A1), bus status (S2-S0) and the direction strobe ( $\overline{MWR}$ ) are driven and do not change for the remainder of the memory cycle. On the falling edge of T1, the SONIC-16 deasserts  $\overline{ECS}$ .  $\overline{ADS}$  is deasserted on the rising edge of T2.

In Synchronous mode,  $\overline{RDYi}$  is sampled on the rising edge at the end of T2 (the rising edge of the next T1 or Tx). T2

states will be repeated until  $\overline{RDYi}$  is sampled properly in a low state.  $\overline{RDYi}$  must meet the setup and hold times with respect to the rising edge of bus clock for proper operation. During read cycles (Figures 5-14), data (D15-D0) is latched at the rising edge at the end of T2. For write cycles (Figure 5-15) data is driven on the falling edge of T1 and stays driven until the end of the cycle.

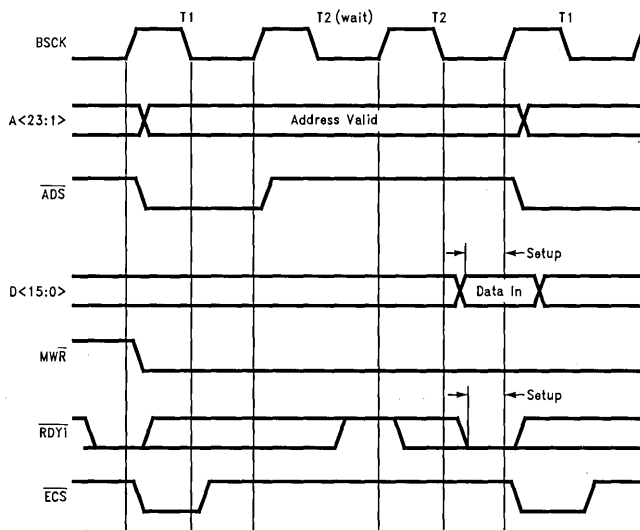


FIGURE 5-14. Memory Read, BMODE = 0, Synchronous (1 Wait-State)

TL/F/11722-38

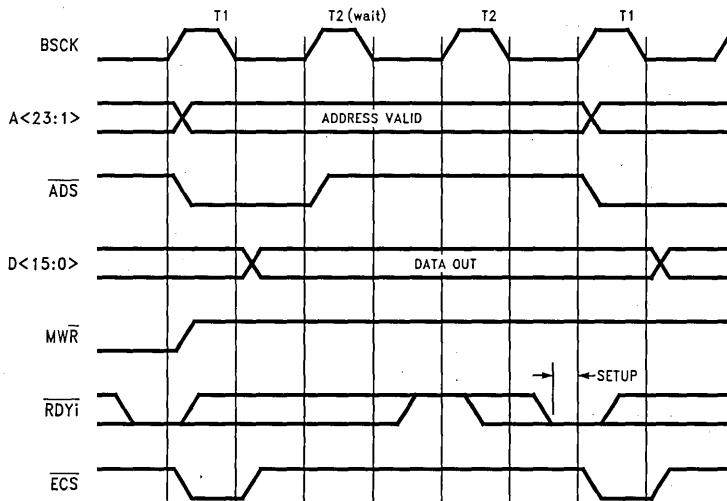


FIGURE 5-15. Memory Write, BMODE = 0, Synchronous (1 Wait-State)

TL/F/11722-40

## 5.0 Bus Interface (Continued)

### 5.4.5.5 Memory Cycle for BMODE = 0, Asynchronous Mode

On the rising edge of T1, the SONIC-16 asserts  $\overline{ADS}$  and  $\overline{ECS}$  to indicate that the memory cycle is starting. The address (A23-A1), bus status (S2-S0) and the direction strobe ( $\overline{MWR}$ ) are driven and do not change for the remainder of the memory cycle. On the falling edge of T1, the SONIC-16 deasserts  $\overline{ECS}$ .  $\overline{ADS}$  is deasserted on the rising edge of T2.

In Asynchronous mode,  $\overline{RDYi}$  is asynchronously sampled on the falling edge of both T1 and T2.  $\overline{RDYi}$  does not need to be synchronized to the bus clock because the chip always resolves these signals to either a high or low state. Meeting the setup time for  $\overline{RDYi}$  guarantees that the SONIC-16 will terminate the memory cycle  $1\frac{1}{2}$  bus clocks after  $\overline{RDYi}$  was sampled. T2 states will be repeated until  $\overline{RDYi}$  is sampled properly in a low state (see note following).

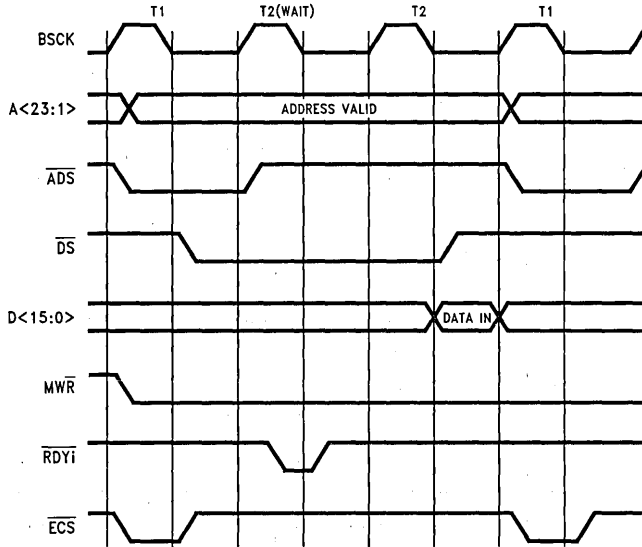


FIGURE 5-16. Memory Read, BMODE = 0, Asynchronous (1 Wait-State)

TL/F/11722-42

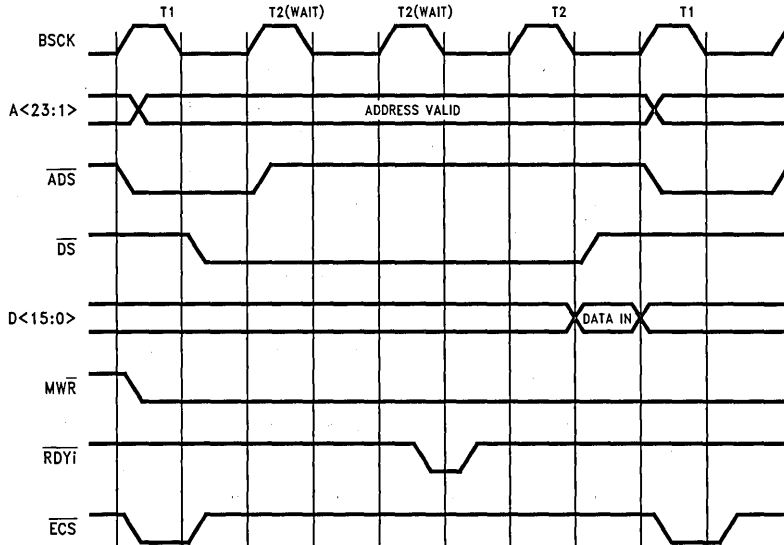


FIGURE 5-17. Memory Read, BMODE = 0, Asynchronous (2 Wait-State)

TL/F/11722-43

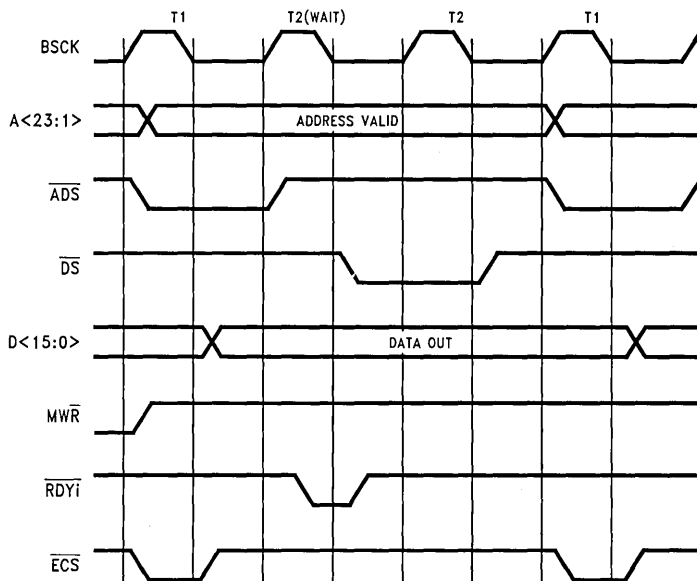


## 5.0 Bus Interface (Continued)

During read cycles (Figures 5-16 and 5-17), data (D15-D0) is latched at the falling edge of T2 and  $\overline{DS}$  is asserted at the falling edge of T1. For write cycles (Figures 5-18 and 5-19) data is driven on the falling edge of T1. If there are wait states inserted,  $\overline{DS}$  is asserted on the falling edge of the first T2(wait).  $\overline{DS}$  is not asserted for zero wait state write cycles.

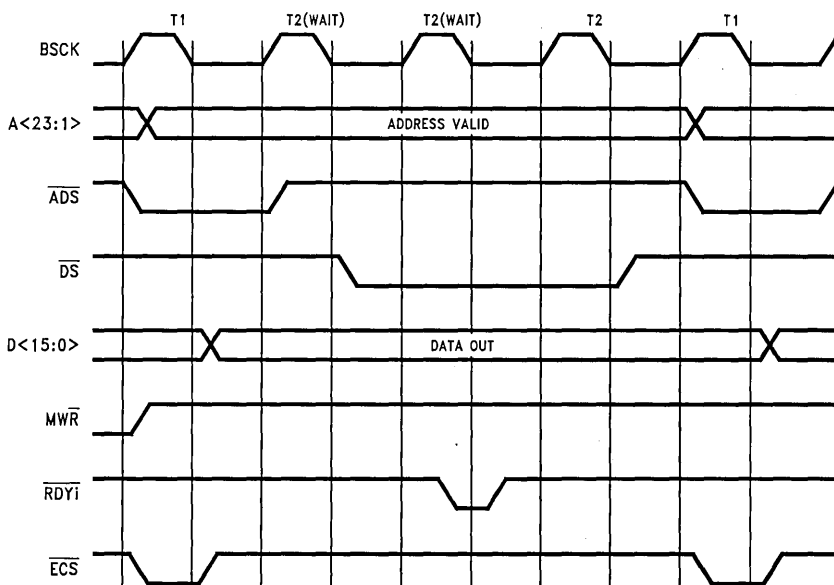
The SONIC-16 terminates the memory cycle by deasserting  $\overline{DS}$  at the falling edge of T2.

**Note:** If the setup time for  $\overline{RDYi}$  is met during T1, the full asynchronous bus cycle will take only 2 bus clocks. This may be an unwanted situation. If so,  $\overline{RDYi}$  should be deasserted during T1.



TL/F/11722-44

FIGURE 5-18. Memory Write, BMODE = 0, Asynchronous (1 Wait-State)



TL/F/11722-45

FIGURE 5-19. Memory Write, BMODE = 0, Asynchronous (2 Wait-State)

## 5.0 Bus Interface (Continued)

### 5.4.6 Bus Exceptions (Bus Retry)

The SONIC-16 provides the capability of handling errors during the execution of the bus cycle (Figure 5-20).

The system asserts  $\overline{BRT}$  (bus retry) to force the SONIC-16 to repeat the current memory cycle. When the SONIC-16 detects the assertion of  $\overline{BRT}$ , it completes the memory cycle at the end of T2 and gets off the bus by deasserting  $\overline{BGACK}$  or  $\overline{HOLD}$ . Then, if Latched Bus Retry mode is not set (LBR in the Data Configuration Register, Section 4.3.2), the SONIC-16 requests the bus again to retry the same memory cycle. If Latched Bus Retry is set, though, the SONIC-16 will not retry until the BR bit in the ISR (see Section 4.3.6) has been reset and  $\overline{BRT}$  is deasserted.  $\overline{BRT}$  has precedence of terminating a memory cycle over  $\overline{DSACK0,1}$ ,  $\overline{STERM}$  or  $\overline{RDYi}$ .

$\overline{BRT}$  may be sampled synchronously or asynchronously by setting the EXBUS bit in the DCR (see Section 4.3.2). If synchronous Bus Retry is set,  $\overline{BRT}$  is sampled on the rising edge of T2. If asynchronous Bus Retry is set,  $\overline{BRT}$  is double synchronized from the falling edge of T1. The asynchronous setup time does not need to be met, but doing so will guarantee that the bus exception will occur in the current bus cycle instead of the next bus cycle. Asynchronous Bus Retry may only be used when the SONIC-16 is set to asynchronous mode.

**Note 1:** The deassertion edge of  $\overline{HOLD}$  is dependent on the PH bit in the DCR2 (see Section 4.3.7). Also,  $\overline{BGACK}$  is driven high for about 1/2 bus clock before going TRI-STATE.

**Note 2:** If Latched Bus retry is set,  $\overline{BRT}$  need only satisfy its setup time (the hold time is not important). Otherwise,  $\overline{BRT}$  must remain asserted until after the Th state.

**Note 3:** If  $\overline{DSACK0,1}$ ,  $\overline{STERM}$  or  $\overline{RDYi}$  remain asserted after  $\overline{BRT}$ , the next memory cycle, may be adversely affected.

### 5.4.7 Slave Mode Bus Cycle

The SONIC-16's internal registers can be accessed by one of two methods (BMODE = 1 or BMODE = 0). In both methods, the SONIC-16 is a slave on the bus. This section describes the SONIC-16's slave mode bus operations.

#### 5.4.7.1 Slave Cycle for BMODE = 1

The system accesses the SONIC-16 by driving  $\overline{SAS}$ ,  $\overline{SRW}$  and  $RA <5:0>$ . These signals will be sampled each bus cycle, but the SONIC-16 will not actually start a slave cycle until  $\overline{CS}$  has also been asserted.  $\overline{CS}$  should not be asserted before  $\overline{SAS}$  is driven low as this will cause improper slave

operation. Once  $\overline{SAS}$  has been driven low, between one and two bus clocks after the assertion of  $\overline{CS}$ ,  $\overline{SMACK}$  will be asserted to signify that the SONIC-16 has started the slave cycle. Although  $\overline{CS}$  is an asynchronous input, meeting its setup time (as shown in Figures 5-21 and 5-22) will guarantee that  $\overline{SMACK}$ , which is asserted off of a falling edge, will be asserted 1 bus clock after the falling edge that  $\overline{CS}$  is clocked in on. This is assuming that the SONIC-16 is not a bus master when  $\overline{CS}$  was asserted. If the SONIC-16 is a bus master, then, when  $\overline{CS}$  is asserted, the SONIC-16 will complete its current master bus cycle and get off the bus temporarily (see Section 5.4.8). In this case,  $\overline{SMACK}$  will be asserted 5 bus clocks after the falling edge that  $\overline{CS}$  was clocked in on. This is assuming that there were no wait states in the current master mode access. Wait states will increase the time for  $\overline{SMACK}$  to go low by the number of wait states in the cycle.

If the slave access is a read cycle (Figure 5-21), then the data will be driven off the same edge as  $\overline{SMACK}$ . If it is a write cycle (Figure 5-22), then the data will be latched in exactly 2 bus clocks after the assertion of  $\overline{SMACK}$ . In either case,  $\overline{DSACK0,1}$  are driven low 2 bus clocks after  $\overline{SMACK}$  to terminate the slave cycle. For a read cycle, the assertion of  $\overline{DSACK0,1}$  indicates valid register data and for a write cycle, the assertion indicates that the SONIC-16 has latched the data. The SONIC-16 deasserts  $\overline{DSACK0,1}$ ,  $\overline{SMACK}$  and the data if the cycle is a read cycle at the rising edge of  $\overline{SAS}$  or  $\overline{CS}$  depending on which is deasserted first.

**Note 1:** Although the SONIC-16 responds as a 32-bit peripheral when it drives  $\overline{DSACK0,1}$  low, it transfers data only on lines D <15:0>.

**Note 2:** For multiple register accesses,  $\overline{CS}$  can be held low and  $\overline{SAS}$  can be used to delimit the slave cycle (this is the only case where  $\overline{CS}$  may be asserted before  $\overline{SAS}$ ). In this case,  $\overline{SMACK}$  will be driven low due to  $\overline{SAS}$  going low since  $\overline{CS}$  has already been asserted. Notice that this means  $\overline{SMACK}$  will not stay asserted low during the entire time  $\overline{CS}$  is low (as is the case for  $\overline{MREQ}$ , Section 5.4.8).

**Note 3:** If memory request ( $\overline{MREQ}$ ) follows a chip select ( $\overline{CS}$ ), it must be asserted at least 2 bus clocks after  $\overline{CS}$  is deasserted. Both  $\overline{CS}$  and  $\overline{MREQ}$  must not be asserted concurrently.

**Note 4:** When  $\overline{CS}$  is deasserted, it must remain deasserted for at least one bus clock.

**Note 5:** The way in which  $\overline{SMACK}$  is asserted due to  $\overline{CS}$  is not the same as the way in which  $\overline{SMACK}$  is asserted due to  $\overline{MREQ}$ . The assertion of  $\overline{SMACK}$  is dependent upon both  $\overline{CS}$  and  $\overline{SAS}$  being low, not just  $\overline{CS}$ . This is not the same as the case for  $\overline{MREQ}$  (see Section 5.4.8). The assertion of  $\overline{SMACK}$  in these two cases should not be confused.

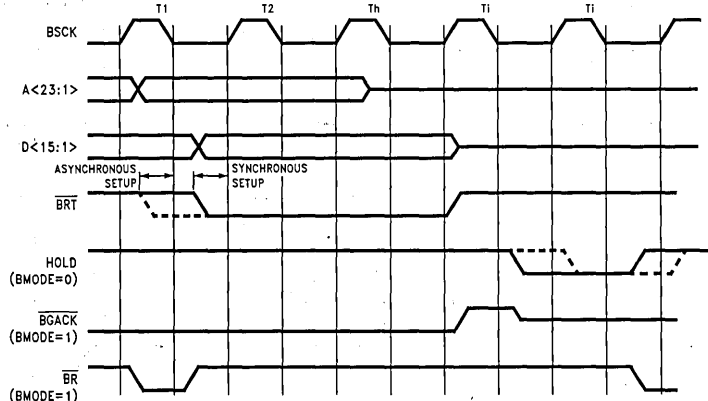


FIGURE 5-20. Bus Exception (Bus Retry)

TL/F/11722-46

5.0 Bus Interface (Continued)

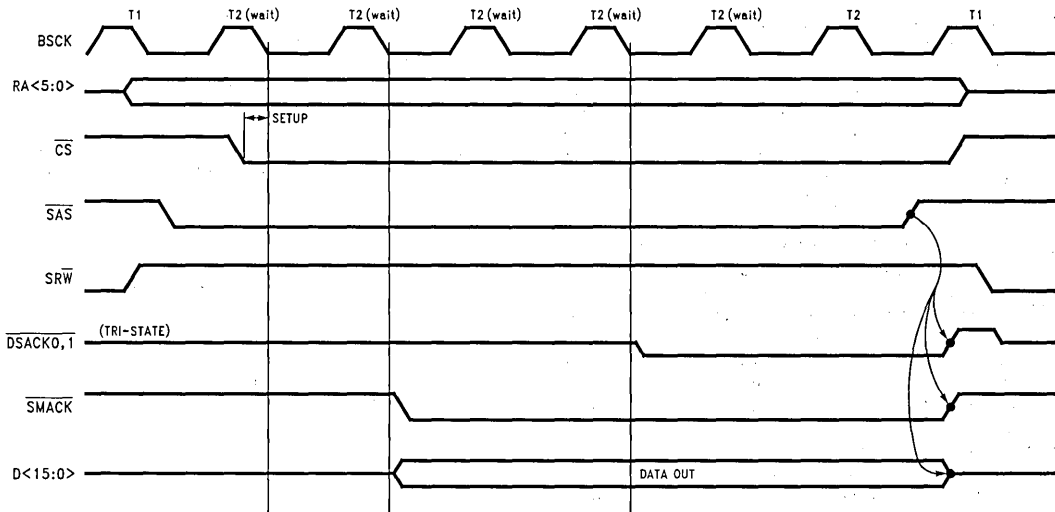


FIGURE 5-21. Register Read, BMODE = 1

TL/F/11722-47

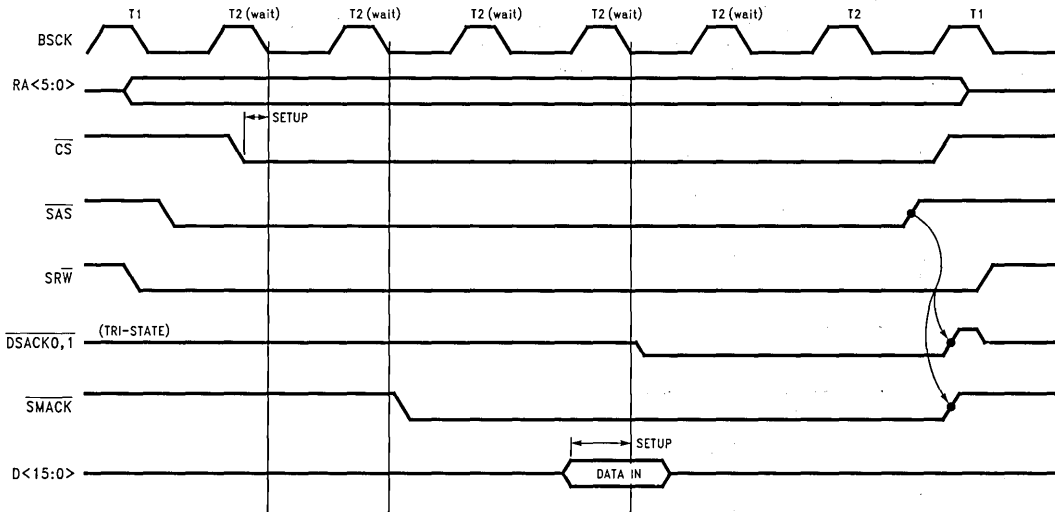


FIGURE 5-22. Register Write, BMODE = 1

TL/F/11722-48

## 5.0 Bus Interface (Continued)

### 5.4.7.2 Slave Cycle for BMODE = 0

The system accesses the SONIC-16 by driving  $\overline{SAS}$ ,  $\overline{CS}$ ,  $\overline{SWR}$  and  $RA<5:0>$ . These signals will be sampled each bus cycle, but the SONIC-16 will not actually start a slave cycle until  $\overline{CS}$  has been sampled low and  $\overline{SAS}$  has been sampled high.  $\overline{CS}$  should not be asserted low before the falling edge of  $\overline{SAS}$  as this will cause improper slave operation.  $\overline{CS}$  may be asserted low, however, before the rising edge of  $\overline{SAS}$ . In this case, it is suggested that  $\overline{SAS}$  be driven high within one bus clock after the falling edge of  $\overline{CS}$ . Between one and two bus clocks after the assertion of  $\overline{CS}$ , once  $\overline{SAS}$  has been driven high,  $\overline{SMACK}$  will be driven low to signify that the SONIC-16 has started the slave cycle. Although  $\overline{CS}$  is an asynchronous input, meeting its setup time (as shown in *Figures 5-23* and *5-24*) will guarantee that  $\overline{SMACK}$ , which is asserted off a falling edge, will be asserted 1 bus clock after the falling edge that  $\overline{CS}$  was clocked in on. This is assuming that the SONIC-16 is not a bus master when  $\overline{CS}$  is asserted. If the SONIC-16 is a bus master, then, when  $\overline{CS}$  is asserted, the SONIC-16 will complete its current master bus cycle and get off the bus temporarily (see Section 5.4.8). In this case,  $\overline{SMACK}$  will be asserted 5 bus clocks after the falling edge that  $\overline{CS}$  was clocked in on. This is assuming that there were no wait states in the current master mode access. Wait states will increase the time for  $\overline{SMACK}$  to go low by the number of wait states in the cycle.

If the slave access is a read cycle (*Figure 5-23*), then the data will be driven off the same edge as  $\overline{SMACK}$ . If it is a write cycle (*Figure 5-24*), then the data will be latched in exactly 2 bus clocks after the assertion of  $\overline{SMACK}$ . In either case,  $\overline{RDY}_0$  is driven low  $2\frac{1}{2}$  bus clocks after  $\overline{SMACK}$  to terminate the slave cycle. For a read cycle, the assertion of  $\overline{RDY}_0$  indicates valid register data and for a write cycle, the assertion indicates that the SONIC-16 has latched the data. The SONIC-16 deasserts  $\overline{RDY}_0$ ,  $\overline{SMACK}$  and the data if the cycle is a read cycle at the falling edge of  $\overline{SAS}$  or the rising edge of  $\overline{CS}$  depending on which is first.

**Note 1:** The SONIC-16 transfers data only on lines  $D<15:0>$  during slave mode accesses.

**Note 2:** For multiple register accesses,  $\overline{CS}$  can be held low and  $\overline{SAS}$  can be used to delimit the slave cycle (this is the only case where  $\overline{CS}$  may be asserted before  $\overline{SAS}$ ). In this case,  $\overline{SMACK}$  will be driven low due to  $\overline{SAS}$  going high since  $\overline{CS}$  has already been asserted. Notice that this means  $\overline{SMACK}$  will not stay asserted low during the entire time  $\overline{CS}$  is low (as is the case for  $\overline{MREQ}$ , Section 5.4.8).

**Note 3:** If memory request ( $\overline{MREQ}$ ) follows a chip select ( $\overline{CS}$ ), it must be asserted at least 2 bus clocks after  $\overline{CS}$  is deasserted. Both  $\overline{CS}$  and  $\overline{MREQ}$  must not be asserted concurrently.

**Note 4:** When  $\overline{CS}$  is deasserted, it must remain deasserted for at least one bus clock.

**Note 5:** The way in which  $\overline{SMACK}$  is asserted due to  $\overline{CS}$  is not the same as the way in which  $\overline{SMACK}$  is asserted due to  $\overline{MREQ}$ . The assertion of  $\overline{SMACK}$  is dependent upon both  $\overline{CS}$  and  $\overline{SAS}$  being low, not just  $\overline{CS}$ . This is not the same as the case for  $\overline{MREQ}$  (see Section 5.4.8). The assertion of  $\overline{SMACK}$  in these two cases should not be confused.

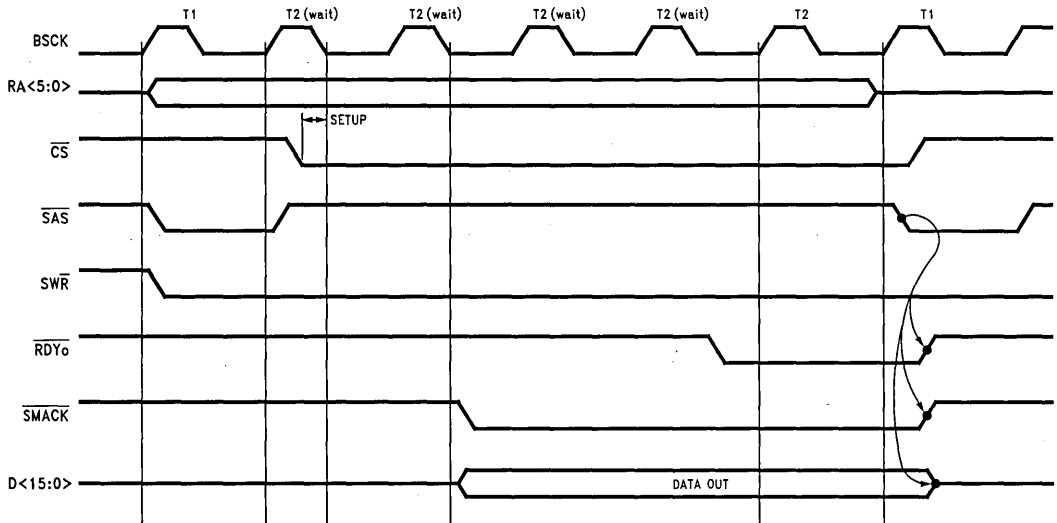


FIGURE 5-23. Register Read, BMODE = 0

TL/F/11722-49

### 5.0 Bus Interface (Continued)

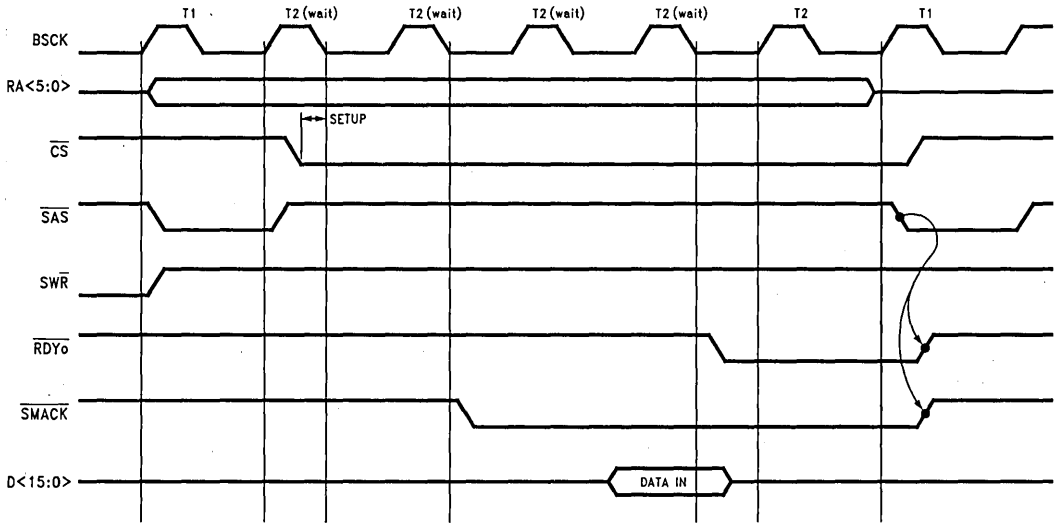


FIGURE 5-24. Register Write, BMODE=0

TL/F/11722-50

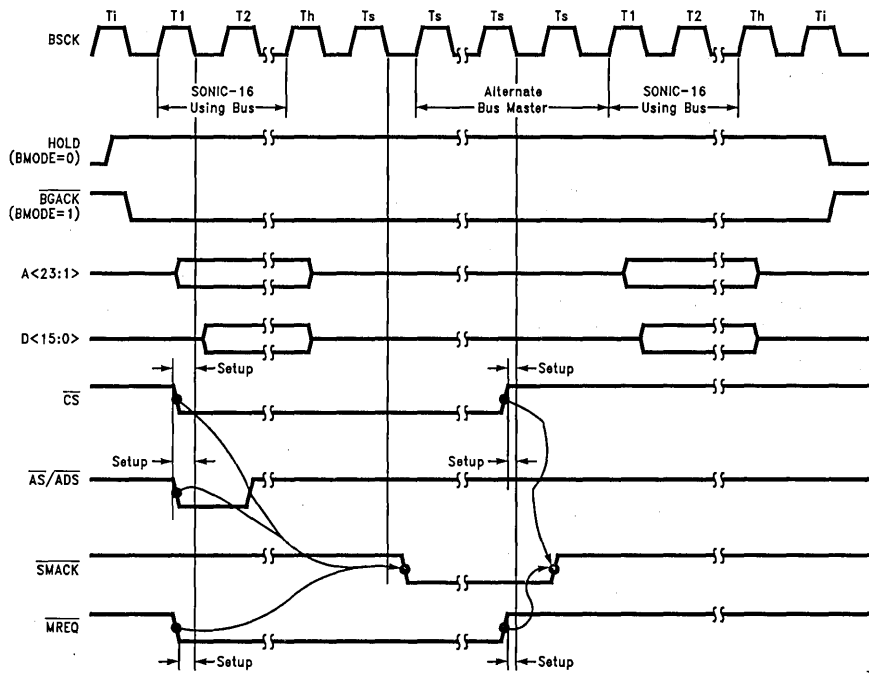


FIGURE 5-25. On-Chip Memory Arbitrer

TL/F/11722-51

## 5.0 Bus Interface (Continued)

### 5.4.8 On-Chip Memory Arbiter

For applications which share the buffer memory area with the host system (shared-memory applications), the SONIC-16 provides a fast on-chip memory arbiter for efficiently resolving accesses between the SONIC-16 and the host system (Figure 5-25). The host system indicates its intentions to use the shared-memory by asserting Memory Request (MREQ). The SONIC-16 will allow the host system to use the shared memory by acknowledging the host system's request with Slave and Memory Acknowledge (SMACK). Once SMACK is asserted, the host system may use the shared memory freely. The host system gives up the shared memory by deasserting MREQ.

MREQ is clocked in on the falling edge of bus clock and is double synchronized internally to the rising edge. SMACK is asserted on the falling edge of a Ts bus cycle. If the SONIC-16 is not currently accessing the memory, SMACK is asserted immediately after MREQ was clocked in. If, however, the SONIC-16 is accessing the shared memory, it finishes its current memory transfer and then issues SMACK. SMACK will be asserted 1 or 5 (see Note 2 below) bus clocks, respectively, after MREQ is clocked in. Since MREQ is double synchronized, it is not necessary to meet its setup time. Meeting the setup time for MREQ will, however, guarantee that SMACK is asserted in the next or fifth bus clock after the current bus clock. SMACK will deassert within one bus clock after MREQ is deasserted. The SONIC-16 will then finish its master operation if it was using the bus previously. If the host system needs to access the SONIC-16's registers instead of shared memory, CS would be asserted instead of MREQ. Accessing the SONIC-16's registers works almost exactly the same as accessing the shared memory except that the SONIC-16 goes into a slave cycle instead of going idle. See Section 5.4.7 for more information about how register accesses work.

**Note 1:** The successive assertion of CS and MREQ must be separated by at least two bus clocks. Both CS and MREQ must not be asserted concurrently.

**Note 2:** The number of bus clocks between MREQ being asserted and the assertion of SMACK when the SONIC-16 is in Master Mode is 5 bus clocks assuming there were no wait states in the Master Mode access. Wait states will increase the time for SMACK to go low by the number of wait states in the cycle (the time will be 5 + the number of wait states).

**Note 3:** The way in which SMACK is asserted due to CS is not the same as the way in which SMACK is asserted due to MREQ. SMACK goes low as a direct result of the assertion of MREQ, whereas, for CS, SAS must also be driven low (BMODE = 1) or high (BMODE = 0) before SMACK will be asserted. This means that when SMACK is asserted due to MREQ, SMACK will remain asserted until MREQ is deasserted. Multiple memory accesses can be made to the shared memory without SMACK ever going high. When SMACK is asserted due to CS, however, SMACK will only remain low as long as SAS is also low (BMODE = 1) or high (BMODE = 0). SMACK will not remain low throughout multiple register accesses to the SONIC-16 because SAS must toggle for each register access. This is an important difference to consider when designing shared memory designs.

TABLE 5-4. Internal Register Content after Reset

Register	Contents after Reset	
	Hardware Reset	Software Reset
Command	0094h	0094h/00A4h
Data Configuration (DCR and DCR2)	*	unchanged
Interrupt Mask	0000h	unchanged
Interrupt Status	0000h	unchanged
Transmit Control	0101h	unchanged
Receive Control	**	unchanged
End Of Buffer Count	02F8h	unchanged
Sequence Counters	0000h	unchanged
CAM Enable	0000h	unchanged

\*Bits 15 and 13 of the DCR and bits 4 through 0 of the DCR2 are reset to a 0 during a hardware reset. Bits 15-12 of the DCR2 are unknown until written to. All other bits in these two registers are unchanged.

\*\*Bits LB1, LB0 and BRD are reset to a 0 during hardware reset. All other bits are unchanged.

### 5.4.9 Chip Reset

The SONIC-16 has two reset modes; a hardware reset and a software reset. The SONIC-16 can be hardware reset by asserting the RESET pin or software reset by setting the RST bit in the Command Register (Section 4.3.1). The two reset modes are not interchangeable since each mode performs a different function.

After power-on, the SONIC-16 must be hardware reset before it will become operational. This is done by asserting RESET for a minimum of 10 transmit clocks (10 Ethernet transmit clock periods, TXC). If the bus clock (BSCK) period is greater than the transmit clock period, RESET should be asserted for 10 bus clocks instead of 10 transmit clocks. A hardware reset places the SONIC-16 in the following state. (The registers affected are listed in parentheses. See Table 5-4 and section 4.3 for more specific information about the registers and how they are affected by a hardware reset. Only those registers listed below and in Table 5-4 are affected by a hardware reset.)

1. Receiver and Transmitter are disabled (CR).
2. The General Purpose timer is halted (CR).
3. All interrupts are masked out (IMR).
4. The NCRS and PTX status bits in the Transmit Control Register (TCR) are set.
5. The End Of Byte Count (EOBC) register is set to 02F8h (760 words).
6. Packet and buffer sequence number counters are set to zero.
7. All CAM entries are disabled. The broadcast address is also disabled (CAM Enable Register and the RCR).
8. Loopback operation is disabled (RCR).
9. The latched bus retry is set to the unlatched mode (DCR).
10. All interrupt status bits are reset (ISR).
11. The Extended Bus Mode is disabled (DCR).
12. HOLD will be asserted/deasserted from the falling clock edge (DCR2).

### 5.0 Bus Interface (Continued)

- 13.  $\overline{PCOMP}$  will not be asserted (DCR2).
- 14. Packets will be accepted (not rejected) on CAM match (DCR2).

A software reset immediately terminates DMA operations and future interrupts. The chip is put into an idle state where registers can be accessed, but the SONIC-16 will not be active in any other way. The registers are affected by a software reset as shown in Table 5-4 (only the Command Register is changed).

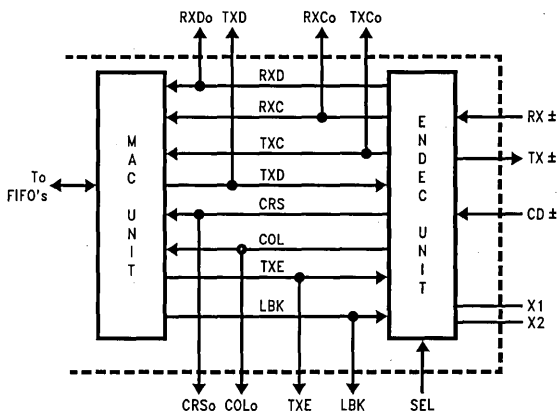
### 6.0 Network Interfacing

The SONIC-16 contains an on-chip ENDEC that performs the network interfacing between the AUI (Attachment Unit

Interface) and the SONIC-16's MAC unit. A pin selectable option allows the internal ENDEC to be disabled and the MAC/ENDEC signals to be supplied to the user for connection to an external ENDEC. If the EXT pin is tied to ground (EXT=0) the internal ENDEC is selected and if EXT is tied to  $V_{CC}$  (EXT=1) the external ENDEC option is selected.

**Internal ENDEC:** When the internal ENDEC is used (EXT=0) the interface signals between the ENDEC and MAC unit are internally connected. While these signals are used internally by the SONIC-16 they are also provided as an output to the user (Figure 6-1).

The internal ENDEC allows for a 2-chip solution for the complete Ethernet interface. Figure 6-2 shows a typical diagram of the network interface.



TL/F/11722-52

FIGURE 6-1. MAC and Internal ENDEC Interface Signals

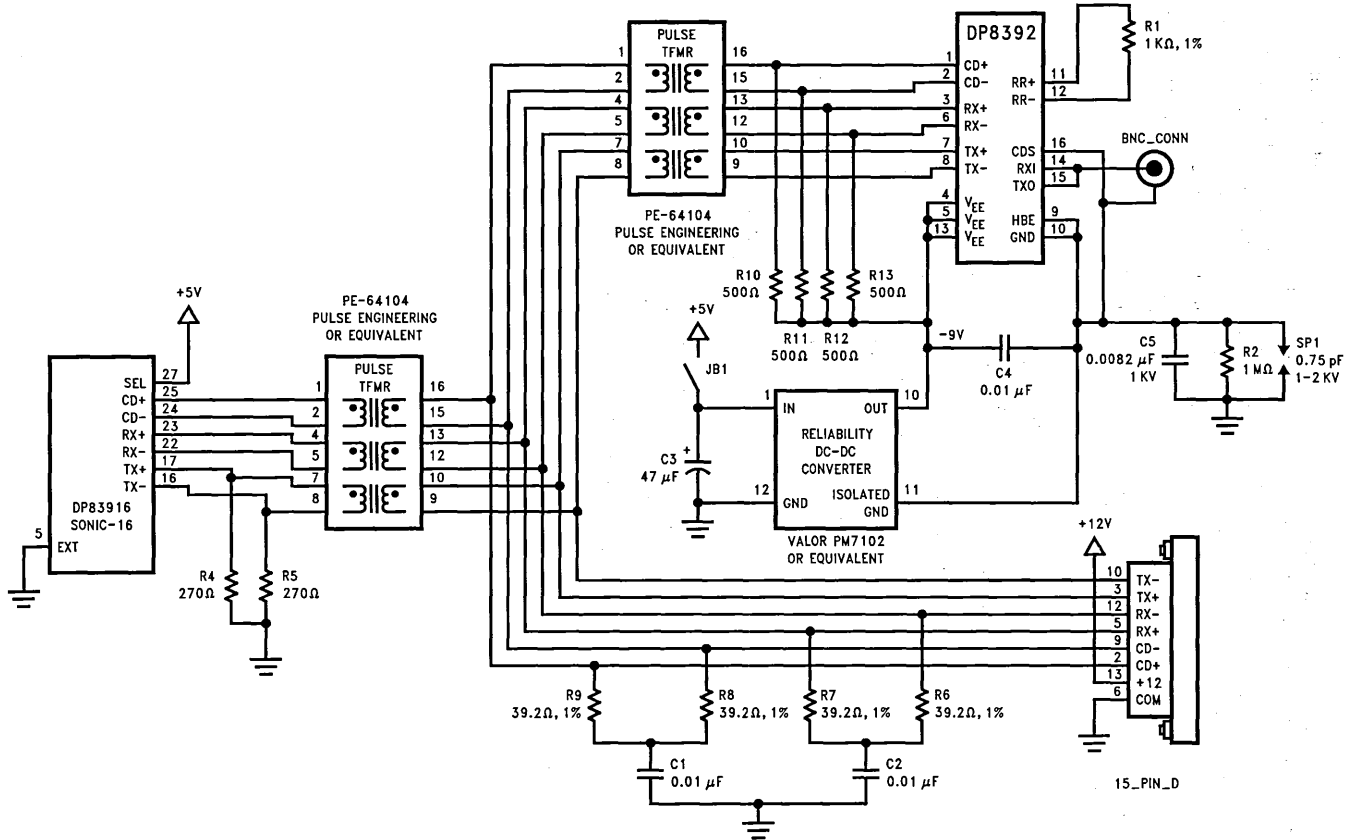


FIGURE 6-2. Network Interface Example (EXT = 0, Using a Single Jumper, JB1, for Network Interface Selection)



## 6.0 Network Interfacing (Continued)

**External ENDEC:** When EXT=1 the internal ENDEC is bypassed and the signals are provided directly to the user. Since SONIC-16's on-chip ENDEC is the same as National's DP83910 Serial Network Interface (SNI) the interface considerations discussed in this section would also apply to using this device in the external ENDEC mode.

### 6.1 MANCHESTER ENCODER AND DIFFERENTIAL DRIVER

The ENDEC unit's encoder begins operation when the MAC section begins sending the serial data stream. It converts NRZ data from the MAC section to Manchester data for the differential drivers (TX+ / -). In Manchester encoding, the first half of the bit cell contains the complementary data and the second half contains the true data (Figure 6-3). A transition always occurs at the middle of the bit cell. As long as the MAC continues sending data, the ENDEC section remains in operation. At the end of transmission, the last transition is always positive, occurring at the center of the bit cell if the last bit is a one, or at the end of the bit cell if the last bit is a zero.

The differential transmit pair drives up to 50 meters of twisted pair AUI cable. These outputs are source followers which require two 270 $\Omega$  pull-down resistors to ground. In addition, a pulse transformer is required between the transmit pair output and the AUI interface.

The driver allows both half-step and full-step modes for compatibility with Ethernet I and IEEE 802.3. When the SEL pin is tied to ground (for Ethernet I), TX+ is positive with respect to TX- during idle on the primary side of the isolation transformer (Figure 6-2). When SEL is tied to V<sub>CC</sub> (for IEEE 802.3), TX+ and TX- are equal in the idle state.

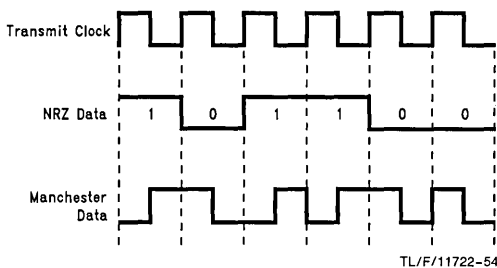


FIGURE 6.3. Manchester Encoded Data Stream

#### 6.1.1 Manchester Decoder

The decoder consists of a differential receiver and a phase lock loop (PLL) to separate the Manchester encoded data stream into clock signals and NRZ data. The differential input must be externally terminated with two 39 $\Omega$  resistors connected in series. In addition, a pulse transformer is required between the receive input pair and the AUI interface.

To prevent noise from falsely triggering the decoder, a squelch circuit at the input rejects signals with a magnitude

less than -175 mV. Signals more negative than -300 mV are decoded.

Once the input exceeds the squelch requirements, the decoder begins operation. The decoder may tolerate bit jitter up to 18 ns in the received data. The decoder detects the end of a frame within one and a half bit times after the last bit of data.

#### 6.1.2 Collision Translator

When the Ethernet transceiver (DP8392 CTI) detects a collision, it generates a 10 MHz signal to the differential collision inputs (CD+ and CD-) of the SONIC-16. When SONIC-16 detects these inputs active, its Collision translator converts the 10 MHz signal to an active collision signal to the MAC section. This signal causes SONIC-16 to abort its current transmission and reschedule another transmission attempt. The collision differential inputs are terminated the same way as the differential receive inputs and a pulse transformer is required between the collision input pair and the AUI interface. The squelch circuitry is also similar, rejecting pulses with magnitudes less than -175 mV.

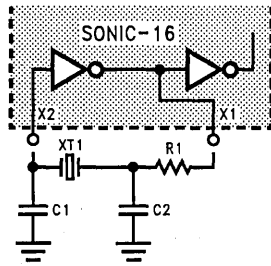
#### 6.1.3 Oscillator Inputs

The oscillator inputs to the SONIC-16 (X1 and X2) can be driven with a parallel resonant crystal or an external clock. In either case the oscillator inputs must be driven with a 20 MHz signal. The signal is divided by 2 to generate the 10 MHz transmit clock (TXC) for the MAC unit. The oscillator also provides internal clock signals for the encoding and decoding circuits.

##### 6.1.3.1 External Crystal

According to the IEEE 802.3 standard, the transmit clock (TXC) must be accurate to 0.01%. This means that the oscillator circuit, which includes the crystal and other parts involved must be accurate to 0.01% after the clock has been divided in half. Hence, when using a crystal, it is necessary to consider all aspects of the crystal circuit. An example of a recommended crystal circuit is shown in Figure 6-4 and suggested oscillator specifications are shown in Table 6-1. The load capacitors in Figure 6-4, C1 and C2, should be no greater than 36 pF each, including all stray capacitance (see note 2 below). The resistor, R1, may be required in order to minimize frequency drift due to changes in V<sub>CC</sub>. If R1 is required, its value must be carefully selected since R1 decreases the loop gain. If R1 is made too large, the loop gain will be greatly reduced and the crystal will not oscillate. If R1 is made too small, normal variations in V<sub>CC</sub> may cause the oscillation frequency to drift out of specification. As a first rule of thumb, the value of R1 should be made equal to five times the motional resistance of the crystal. The motional resistance of 20 MHz crystals is usually in the range of 10 $\Omega$  to 30 $\Omega$ . This implies that reasonable values for R1 should be in the range of 50 $\Omega$  to 150 $\Omega$ . The decision of whether or not to include R1 should be based upon measured variations of crystal frequency as each of the circuit parameters are varied.

## 6.0 Network Interfacing (Continued)



TL/F/11722-55

**FIGURE 6.4. Crystal Connection to the SONIC-16** (see text)

**Note 1:** The X1 pin is not guaranteed to provide a TTL compatible logic output, and should not be used to drive any external logic. If additional logic needs to be driven, then an external oscillator should be used as described in the following section.

**Note 2:** The frequency marked on the crystal is usually measured with a fixed load capacitance specified in the crystal's data sheet. The actual load capacitance used should be the specified value minus the stray capacitance.

**TABLE 6-1. Crystal Specifications**

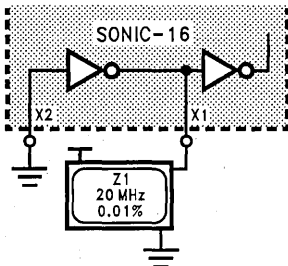
Resonant frequency	20 MHz
Tolerance (see text)	$\pm 0.01\%$ at 25°C
Accuracy	$\pm 0.005\%$ (50 ppm) at 0 to 70°C
Fundamental Mode Series Resistance	$\leq 25\Omega$
Specified Load Capacitance	$\leq 18$ pF
Type	AT cut
Circuit	Parallel Resonance

### 6.1.3.2 Clock Oscillator Module

If an external clock oscillator is used, the SONIC-16 can be connected to the external oscillator in one of two ways. The first configuration is shown in *Figure 6-5*. In this case, an oscillator that provides the following should be used:

1. TTL or CMOS output with a 0.01% frequency tolerance
2. 40%–60% duty cycle
3.  $\geq 5$  TTL loads output drive ( $I_{OL} = 8$  mA) (Additional output drive may be necessary if the oscillator must also drive other components.)

Again, the above assumes no other circuitry is driven.

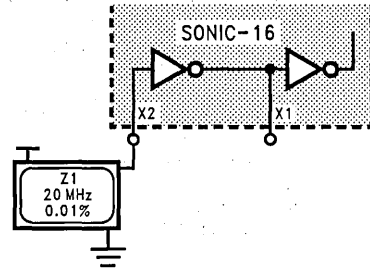


TL/F/11722-56

**FIGURE 6.5. Oscillator Module Connection to the SONIC-16**

The second configuration, shown in *Figure 6-6*, connects to the X2 input. This connection requires an oscillator with the same specifications as the previous circuit except that the

output drive specification need only be one CMOS load. This circuit configuration also offers the advantage of slightly lower power consumption. In this configuration, the X1 pin must be left open and should not drive external circuitry. Also, as shown by *Figure 6-6*, there is a 180° phase difference between connecting an oscillator to X1 compared to X2. This difference only affects the relationship between TXC and the oscillator module output. The operation of the SONIC-16 is not affected by this phase change.



TL/F/11722-57

**FIGURE 6.6. Alternate Oscillator Module Connection to the SONIC-16**

### 6.1.3.3 PCB Layout Considerations

Care should be taken when connecting a crystal. Stray capacitance (e.g., from PC board traces and plated through holes around the X1 and X2 pins) can shift the crystal's frequency out of range, causing the transmitted frequency to exceed the 0.01% tolerance specified by IEEE. The layout considerations for using an external crystal are rather straightforward. The oscillator layout should locate all components close to the X1 and X2 pins and should use short traces that avoid excess capacitance and inductance. A solid ground should be used to connect the ground legs of the two capacitors.

When connecting an external oscillator, the only considerations are to keep the oscillator module as close to the SONIC-16 as possible to reduce stray capacitance and inductance and to give the module a clean  $V_{CC}$  and a solid ground.

### 6.1.4 Power Supply Considerations

In general, power supply routing and design for the SONIC-16 need only follow standard practices. In some situations, however, additional care may be necessary in the layout of the analog supply. Specifically special care may be needed for the TXVCC, RXVCC and PLLVCC power supplies and the TXGND and ANGND. In most cases the analog and digital power supplies can be interconnected. However, to ensure optimum performance of the SONIC-16's analog functions, power supply noise should be minimized. To reduce analog supply noise, any of several techniques can be used.

1. Route analog supplies as a separate set of traces or planes from the digital supplies with their own decoupling capacitors.
2. Provide noise filtering on the analog supply pins by inserting a low pass filter. Alternatively, a ferrite bead could be used to reduce high frequency power supply noise.
3. Utilize a separate regulator to generate the analog supply.

## 7.0 AC and DC Specifications

### Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage ( $V_{CC}$ )	-0.5V to 7.0V
DC Input Voltage ( $V_{IN}$ )	-0.5V to $V_{CC} + 0.5V$
DC Output Voltage ( $V_{OUT}$ )	-0.5V to $V_{CC} + 0.5V$

Storage Temperature Range ( $T_{STG}$ )	-65°C to 150°C
Power Dissipation (PD)	500 mW
Lead Temp. (TL) (Soldering, 10 sec.)	260°C
ESD Rating ( $R_{ZAP} = 1.5k, C_{ZAP} = 120 pF$ )	1.5 KV

### DC Specifications $T_A = 0^\circ\text{C}$ to $70^\circ\text{C}$ , $V_{CC} = 5V \pm 5\%$ unless otherwise specified

Symbol	Parameter	Conditions	Min	Max	Units
$V_{OH}$	Minimum High Level Output Voltage	$I_{OH} = -8 \text{ mA}$	3.0		V
$V_{OL}$	Maximum Low Level Output Voltage	$I_{OL} = 8 \text{ mA}$		0.4	V
$V_{IH}$	Minimum High Level Input Voltage		2.0		V
$V_{IL}$	Maximum Low Level Input Voltage			0.8	V
$I_{IN}$	Input Current	$V_{IN} = V_{CC}$ or GND	-1.0	1.0	$\mu\text{A}$
$I_{OZ}$	Maximum TRI-STATE Output Leakage Current	$V_{OUT} = V_{CC}$ or GND	-10	10	$\mu\text{A}$
$I_{CC}$	Average Operating Supply Current	$I_{OUT} = 0 \text{ mA}$ , Freq = $f_{max}$		80	mA

### AUI INTERFACE PINS ( $TX \pm$ , $RX \pm$ , and $CD \pm$ )

$V_{OD}$	Diff. Output Voltage ( $TX \pm$ )	78 $\Omega$ Termination, and 270 $\Omega$ from Each to GND	$\pm 550$	$\pm 1200$	mV
$V_{OB}$	Diff. Output Voltage Imbalance ( $TX \pm$ )	78 $\Omega$ Termination, and 270 $\Omega$ from Each to GND	Typical: 40 mV		
$V_U$	Undershoot Voltage ( $TX \pm$ )	78 $\Omega$ Termination, and 270 $\Omega$ from Each to GND	Typical: 80 mV		
$V_{DS}$	Diff. Squelch Threshold ( $RX \pm$ and $CD \pm$ )		-175	-300	mV

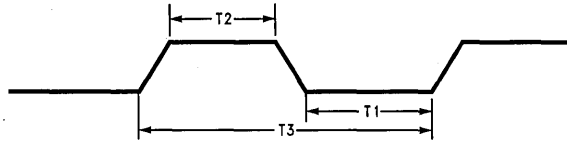
### OSCILLATOR PINS (X1 AND X2)

$V_{IH}$	X1 Input High Voltage	X1 is Connected to an Oscillator and X2 is Grounded	2.0		V
$V_{IL}$	X1 Input Low Voltage	X1 is Connected to an Oscillator and X2 is Grounded		0.8	V
$I_{OSC1}$	X1 Input Current	X1 is Connected to an Oscillator and X2 is Grounded $V_{IN} = V_{CC}$ or GND		8	mA
$V_{IH}$	X2 Input High Voltage	X2 is Connected to an Oscillator and X1 is Open	2.0		V
$V_{IL}$	X2 Input Low Voltage	X2 is Connected to an Oscillator and X1 is Open		0.8	V
$I_{OSC2}$	X2 Input Leakage Current	X2 is Connected to an Oscillator and X1 is Open $V_{IN} = V_{CC}$ or GND	-10	10	$\mu\text{A}$

## 7.0 AC and DC Specifications (Continued)

### AC Specifications

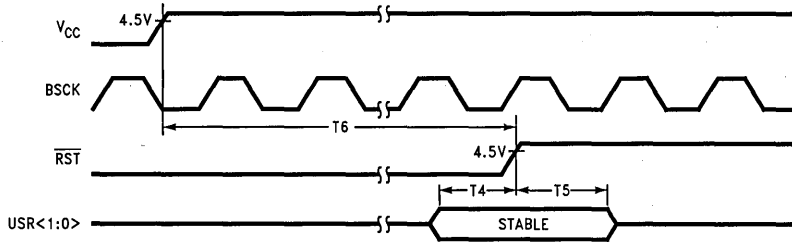
#### BUS CLOCK TIMING



TL/F/11722-58

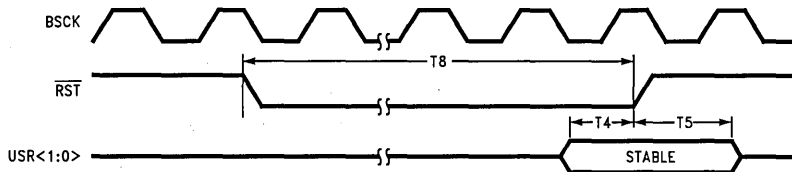
Number	Parameter	20 MHz		Units
		Min	Max	
T1	Bus Clock Low Time	22.5		ns
T2	Bus Clock High Time	22.5		ns
T3	Bus Clock Cycle Time (Note 2)	50	100	ns

#### POWER-ON RESET



TL/F/11722-59

#### NON POWER-ON RESET



TL/F/11722-60

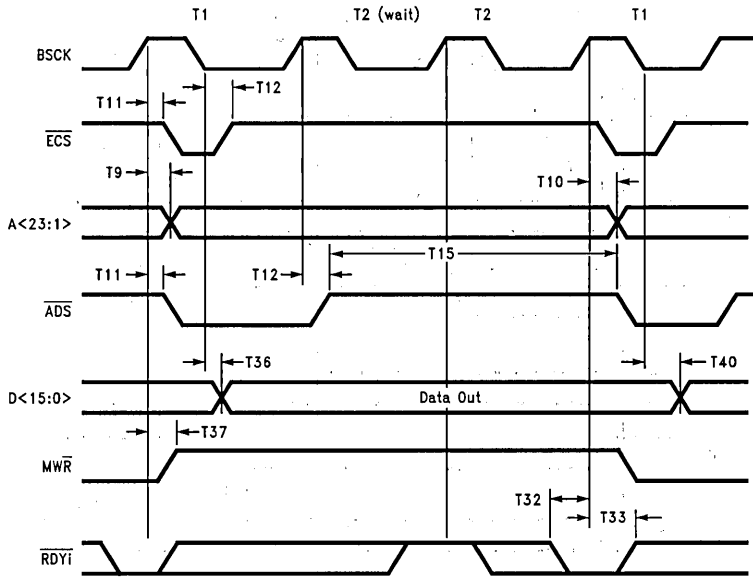
Number	Parameter	20 MHz		Units
		Min	Max	
T4	USR <1:0> Setup to $\overline{RST}$	10		ns
T5	USR <1:0> Hold from $\overline{RST}$	20		ns
T6	Power-On Reset High (Notes 1, 2)	10		TXC
T8	Reset Pulse Width (Notes 1, 2)	10		TXC

**Note 1:** The reset time is determined by the slower of BSCCK or TXC. If BSCCK > TXC, T6 and T8 equal 10 TXCs. If BSCCK < TXC, T6 and T8 equal 10 BSCCKs (T3).

**Note 2:** These specifications are not tested.

## 7.0 AC and DC Specifications (Continued)

MEMORY WRITE, BMODE = 0, SYNCHRONOUS MODE (one wait-state shown)



TL/F/11722-61

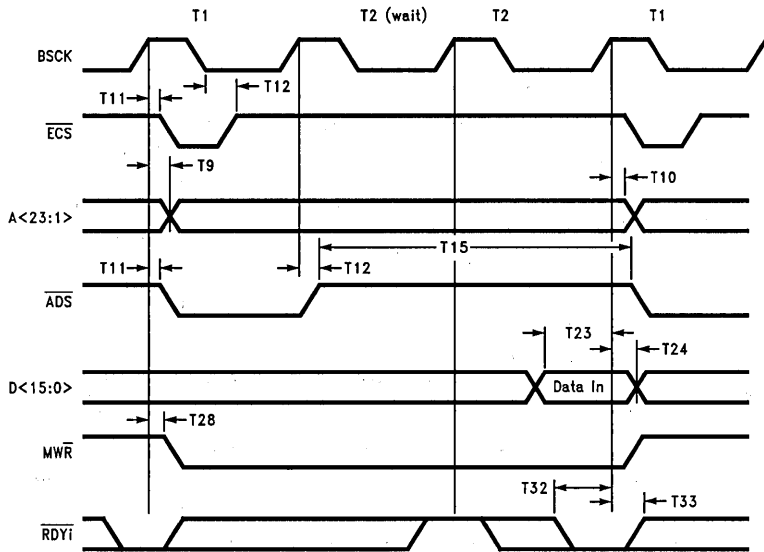
Number	Parameter	20 MHz		Units
		Min	Max	
T9	BSCCK to Address Valid		34	ns
T10	Address Hold Time from BSCCK	5		ns
T11	BSCCK to $\overline{ADS}$ , $\overline{ECS}$ Low		34	ns
T12	BSCCK to $\overline{ADS}$ , $\overline{ECS}$ High		34	ns
T15	$\overline{ADS}$ High Width (Note 2)	bcyc-5		ns
T32	$\overline{RDYi}$ Setup to BSCCK	30		ns
T33	$\overline{RDYi}$ Hold from BSCCK	5		ns
T36	BSCCK to Memory Write Data Valid		70	ns
T37	BSCCK to $\overline{MWR}$ (Write) Valid (Note 1)		30	ns
T40	Write Data Hold Time from BSCCK	10		ns

**Note 1:** For successive read operations,  $\overline{MWR}$  remains low, and for successive write operations,  $\overline{MWR}$  remains high during a transfer. During RDA and TDA transfers the  $\overline{MWR}$  signal will stay either high or low for the entire burst of the transfer. During RDA and TDA transfers the  $\overline{MWR}$  signal will switch on the rising edge of a  $T_i$  (idle) state that is inserted between the read and the write operation.

**Note 2:** bcyc = bus clock cycle time (T3).

## 7.0 AC and DC Specifications (Continued)

MEMORY READ, BMODE = 0, SYNCHRONOUS MODE (one wait-state shown)



TL/F/11722-62

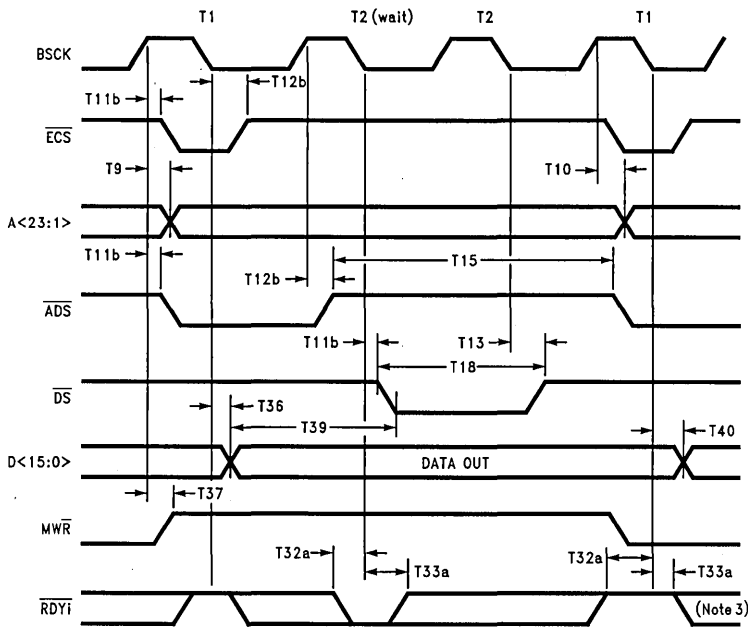
Number	Parameter	20 MHz		Units
		Min	Max	
T9	BSCCK to Address Valid		34	ns
T10	Address Hold Time from BSCCK	5		ns
T11	BSCCK to $\overline{ADS}$ , $\overline{ECS}$ Low		34	ns
T12	BSCCK to $\overline{ADS}$ , $\overline{ECS}$ High		34	ns
T15	$\overline{ADS}$ High Width (Note 2)	bcyc - 5		ns
T23	Read Data Setup Time to BSCCK	12		ns
T24	Read Data Hold Time from BSCCK	7		ns
T28	BSCCK to $\overline{MWR}$ (Read) Valid (Note 1)		30	ns
T32	$\overline{RDYi}$ Setup Time to BSCCK	30		ns
T33	$\overline{RDYi}$ Hold Time to BSCCK	5		ns

**Note 1:** For successive read operations,  $\overline{MWR}$  remains low, and for successive write operations,  $\overline{MWR}$  remains high. During RBA and TBA transfers the  $\overline{MWR}$  signal will stay either high or low for the entire burst of the transfer. During RDA and TDA transfers, the  $\overline{MWR}$  signal will switch on the rising edge of a T1 (idle) state that is inserted between the read and the write operation.

**Note 2:** bcyc = bus clock cycle time (T3).

## 7.0 AC and DC Specifications (Continued)

### MEMORY WRITE, BMODE = 0, ASYNCHRONOUS MODE



TL/F/11722-63

Number	Parameter	20 MHz		Units
		Min	Max	
T9	BSCCK to Address Valid		34	ns
T10	Address Hold Time from BSCCK	5		ns
T11b	BSCCK to $\overline{ADS}$ , $\overline{DS}$ , $\overline{ECS}$ Low		30	ns
T12b	BSCCK to $\overline{ADS}$ , $\overline{ECS}$ High		32	ns
T13	BSCCK to $\overline{DS}$ High		36	ns
T15	$\overline{ADS}$ High Width (Note 2)	bcyc - 5		ns
T18	Write Data Strobe Low Width (Notes 2, 4)	bcyc - 5		ns
T32a	Ready Asynch. Setup to BSCCK (Note 3)	8		ns
T33a	Ready Asynch. Hold from BSCCK	5		ns
T36	BSCCK to Memory Write Data Valid		70	ns
T37	BSCCK to $\overline{MWR}$ (Write) Valid (Note 1)		30	ns
T39	Write Data Valid to Data Strobe Low (Note 2)	bcyc - 40		ns
T40	Write Data Hold Time from BSCCK	10		ns

**Note 1:** For successive read operations,  $\overline{MWR}$  remains low, and for successive write operations,  $\overline{MWR}$  remains high. During RBA and TBA transfers the  $\overline{MWR}$  signal will stay either high or low for the entire burst of the transfer. During RDA and TDA transfers, the  $\overline{MWR}$  signal will switch on the rising edge of a T1 (idle) state that is inserted between the read and the write operation.

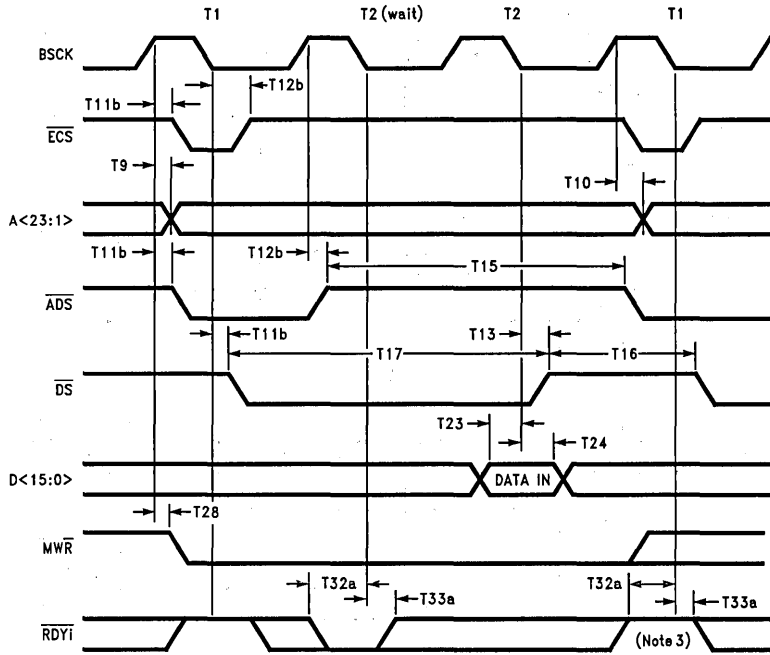
**Note 2:** bcyc = bus clock cycle time (T3)

**Note 3:** This setup time assures that the SONIC-16 terminates the memory cycle on the next bus clock (BSCCK).  $\overline{RDYi}$  does not need to be synchronized to the bus clock, though, since it is an asynchronous input in this case.  $\overline{RDYi}$  is sampled during the falling edge of BSCCK. If the SONIC-16 samples  $\overline{RDYi}$  low during the T1 cycle, the SONIC-16 will finish the current access in a total of two bus clocks instead of three, which would be the case if  $\overline{RDYi}$  had been sampled low during T2(wait). (This is assuming that programmable wait states are set to 0).

**Note 4:**  $\overline{DS}$  will only be asserted if the bus cycle has at least one wait state inserted.

## 7.0 AC and DC Specifications (Continued)

### MEMORY READ, BMODE = 0, ASYNCHRONOUS MODE



TL/F/11722-64

Number	Parameter	20 MHz		Units
		Min	Max	
T9	BSClk to Address Valid		34	ns
T10	Address Hold Time from BSClk	5		ns
T11b	BSClk to $\overline{ADS}$ , $\overline{DS}$ , $\overline{ECS}$ Low		30	ns
T12b	BSClk to $\overline{ADS}$ , $\overline{DS}$ , $\overline{ECS}$ High		32	ns
T13	BSClk to $\overline{DS}$ High		36	ns
T15	$\overline{ADS}$ High Width (Note 2)	bcyc - 5		ns
T16	Read Data Strobe High Width (Note 2)	bcyc - 12		ns
T17	Read Data Strobe Low Width (Note 2)	bcyc - 5		ns
T23	Read Data Setup Time to BSClk	12		ns
T24	Read Data Hold Time from BSClk	7		ns
T28	BSClk to $\overline{MWR}$ (Read) Valid (Note 1)		30	ns
T32a	Ready Asynch. Setup Time to BSClk (Note 3)	8		ns
T33a	Ready Asynch. Hold Time to BSClk	5		ns

**Note 1:** For successive read operations,  $\overline{MWR}$  remains low, and for successive write operations,  $\overline{MWR}$  remains high. During RBA and TBA transfers the  $\overline{MWR}$  signal will stay either high or low for the entire burst of the transfer. During RDA and TDA transfers, the  $\overline{MWR}$  signal will switch on the rising edge of a T1 (idle) state that is inserted between the read and the write operation.

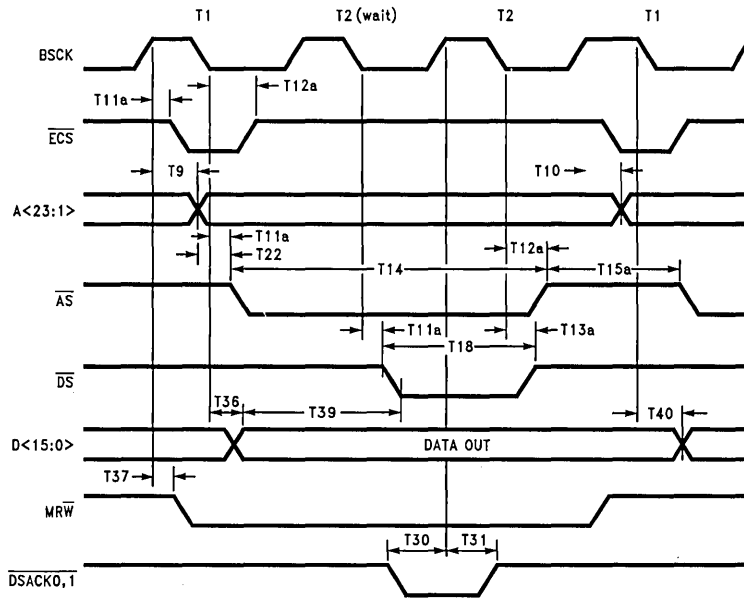
**Note 2:** bcyc = bus clock cycle time (T3)

**Note 3:** This setup time assures that the SONIC-16 terminates the memory cycle on the next bus clock (BSClk).  $\overline{RDYi}$  does not need to be synchronized to the bus clock, though, since it is an asynchronous input in this case.  $\overline{RDYi}$  is sampled during the falling edge of BSClk. If the SONIC-16 samples  $\overline{RDYi}$  low during the T1 cycle, the SONIC-16 will finish the current access in a total of two bus clocks instead of three, which would be the case if  $\overline{RDYi}$  had been sampled low during T2(wait). (This is assuming that programmable wait states are set to 0).



## 7.0 AC and DC Specifications (Continued)

MEMORY WRITE, BMODE = 1, SYNCHRONOUS MODE (one wait-state shown)



TL/F/11722-65

Number	Parameter	20 MHz		Units
		Min	Max	
T9	BSCCK to Address Valid		34	ns
T10	Address Hold Time from BSCCK	5		ns
T11a	BSCCK to $\overline{AS}$ , $\overline{DS}$ , $\overline{ECS}$ Low		26	ns
T12a	BSCCK to $\overline{AS}$ , $\overline{ECS}$ High		34	ns
T13a	BSCCK to $\overline{DS}$ High		36	ns
T14	$\overline{AS}$ Strobe Low Width (Note 3)	bcyc - 7		ns
T15a	$\overline{AS}$ Strobe High Width (Note 3)	bcyc - 15		ns
T18	Write Data Strobe Low Width (Notes 1, 3)	bcyc - 5		ns
T22	Address Valid to $\overline{AS}$ (Note 3)	bch - 18		ns
T30	$\overline{DSACK0,1}$ Setup to BSCCK (Note 4)	8		ns
T31	$\overline{DSACK0,1}$ Hold from BSCCK	12		ns
T36	BSCCK to Memory Write Data Valid		70	ns
T37	BSCCK to $\overline{MRW}$ (Write) Valid (Note 2)		30	ns
T39	Write Data Valid to Data Strobe Low (Note 3)	bcyc - 40		ns
T40	Memory Write Data Hold Time from BSCCK	10		ns

**Note 1:**  $\overline{DS}$  will only be asserted if the bus cycle has at least one wait state inserted.

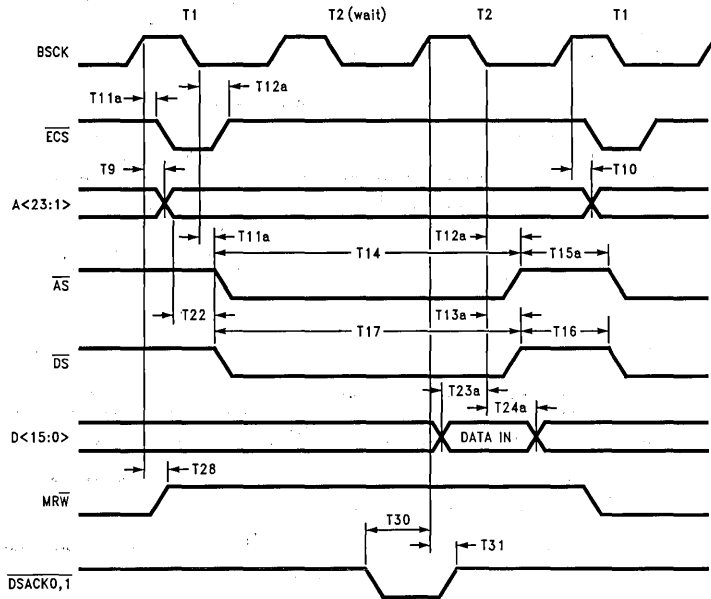
**Note 2:** For successive read operations,  $\overline{MRW}$  remains low, and for successive write operations,  $\overline{MRW}$  remains high. During RBA and TBA transfers the  $\overline{MRW}$  signal will stay either high or low for the entire burst of the transfer. During RDA and TDA transfers, the  $\overline{MRW}$  signal will switch on the rising edge of a  $T_i$  (idle) state that is inserted between the read and the write operation.

**Note 3:** bcyc = bus clock cycle time (T3). bch = bus clock high time (T2).

**Note 4:**  $\overline{DSACK0,1}$  must be synchronized to the bus clock (BSCCK) during synchronous mode.

## 7.0 AC and DC Specifications (Continued)

MEMORY READ, BMODE = 1, SYNCHRONOUS MODE (one wait-state shown)



TL/F/11722-66

Number	Parameter	20 MHz		Units
		Min	Max	
T9	BSCCK to Address Valid		34	ns
T10	Address Hold Time from BSCCK	5		ns
T11a	BSCCK to $\overline{AS}$ , $\overline{DS}$ , $\overline{ECS}$ Low		26	ns
T12a	BSCCK to $\overline{AS}$ , $\overline{ECS}$ High		34	ns
T13a	BSCCK to $\overline{DS}$ High		36	ns
T14	$\overline{AS}$ Strobe Low Width (Note 3)	bcyc - 7		ns
T15a	$\overline{AS}$ Strobe High Width (Note 3)	bcyc - 15		ns
T16	Read Data Strobe High Width (Note 3)	bcyc - 12		ns
T17	Read Data Strobe Low Width (Note 3)	bcyc - 5		ns
T22	Address Valid to $\overline{AS}$ (Note 3)	bch - 18		ns
T23a	Read Data Setup Time to BSCCK	5		ns
T24a	Read Data Hold Time from BSCCK	5		ns
T28	BSCCK to $\overline{MRW}$ (Read) Valid (Note 1)		30	ns
T30	$\overline{DSACK0,T}$ Setup to BSCCK (Note 2)	8	ns	
T31	$\overline{DSACK0,T}$ Hold from BSCCK	12		ns

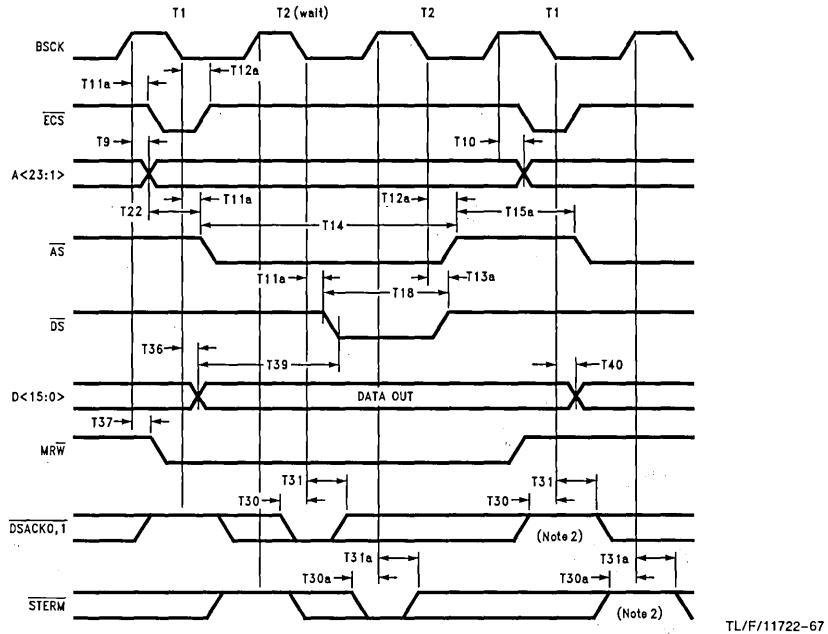
**Note 1:** For successive read operations,  $\overline{MRW}$  remains low, and for successive write operations,  $\overline{MRW}$  remains high. During RBA and TBA transfers the  $\overline{MRW}$  signal will stay either high or low for the entire burst of the transfer. During RDA and TDA transfers, the  $\overline{MRW}$  signal will switch on the rising edge of a Ti (idle) state that is inserted between the read and the write operation.

**Note 2:**  $\overline{DSACK0,T}$  must be synchronized to the bus clock (BSCCK) during synchronous mode.

**Note 3:** bcyc = bus clock cycle time (T3). bch = bus clock high time (T2)

## 7.0 AC and DC Specifications (Continued)

### MEMORY WRITE, BMODE = 1, ASYNCHRONOUS MODE



Number	Parameter	20 MHz		Units
		Min	Max	
T9	BSCCK to Address Valid		34	ns
T10	Address Hold Time from BSCCK	5		ns
T11a	BSCCK to $\overline{AS}$ , $\overline{DS}$ , $\overline{ECS}$ Low		26	ns
T12a	BSCCK to $\overline{AS}$ , $\overline{ECS}$ High		34	ns
T13a	BSCCK to $\overline{DS}$ High		36	ns
T14	$\overline{AS}$ Strobe Low Width (Note 3)	bcyc - 7		ns
T15a	$\overline{AS}$ Strobe High Width (Note 3)	bcyc - 15		ns
T18	Write Data Strobe Low Width (Notes 3, 4)	bcyc - 5		ns
T22	Address Valid to $\overline{AS}$ (Note 3)	bch - 18		ns
T30	$\overline{DSACK0,1}$ Setup to BSCCK (Note 2)	8		ns
T30a	$\overline{STERM}$ Setup to BSCCK (Note 2)	6		ns
T31	$\overline{DSACK0,1}$ Hold from BSCCK	12		ns
T31a	$\overline{STERM}$ Hold from BSCCK	12		ns
T36	BSCCK to Memory Write Data Valid		70	ns
T37	BSCCK to $\overline{MRW}$ (Write) Valid (Note 1)		30	ns
T39	Write Data Valid to Data Strobe Low (Note 3)	bcyc - 40		ns
T40	Memory Write Data Hold from BSCCK	10		ns

**Note 1:** For successive read operations,  $\overline{MRW}$  remains low, and for successive write operations,  $\overline{MRW}$  remains high. During RBA and TBA transfers the  $\overline{MRW}$  signal will stay either high or low for the entire burst of the transfer. During RDA and TDA transfers, the  $\overline{MRW}$  signal will switch on the rising edge of a T1 (idle) state that is inserted between the read and the write operation.

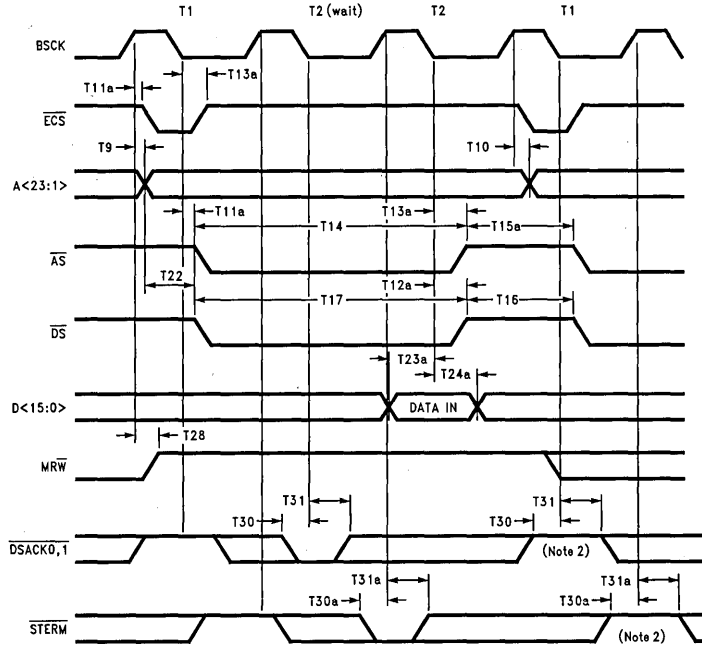
**Note 2:** Meeting the setup time for  $\overline{DSACK0,1}$  or  $\overline{STERM}$  guarantees that the SONIC-16 will terminate the memory cycle  $1\frac{1}{2}$  bus clocks after  $\overline{DSACK0,1}$  were sampled, or 1 cycle after  $\overline{STERM}$  was sampled. T2 states will be repeated until  $\overline{DSACK0,1}$  or  $\overline{STERM}$  are sampled properly in a low state. If the SONIC-16 samples  $\overline{DSACK0,1}$  or  $\overline{STERM}$  low during the T1 or first T2 state respectively, the SONIC-16 will finish the current access in a total of two bus clocks instead of three (assuming that programmable wait states are set to 0).  $\overline{DSACK0,1}$  are asynchronously sampled and  $\overline{STERM}$  is synchronously sampled.

**Note 3:** bcyc = bus clock cycle time (T3). bch = bus clock high time (T2).

**Note 4:**  $\overline{DS}$  will only be asserted if the bus cycle has at least one wait state inserted.

## 7.0 AC and DC Specifications (Continued)

### MEMORY READ, BMODE = 1, ASYNCHRONOUS MODE



TL/F/11722-68

Number	Parameter	20 MHz		Units
		Min	Max	
T9	BSCCK to Address Valid		34	ns
T10	Address Hold Time from BSCCK	5		ns
T11a	BSCCK to $\overline{AS}$ , $\overline{DS}$ , $\overline{ECS}$ Low		26	ns
T12a	BSCCK to $\overline{AS}$ , $\overline{ECS}$ High		34	ns
T13a	BSCCK to $\overline{DS}$ High		36	ns
T14	$\overline{AS}$ Strobe Low Width (Note 3)	bcyc - 7		ns
T15a	$\overline{AS}$ Strobe High Width (Note 3)	bcyc - 15		ns
T16	Read Data Strobe High Width (Note 3)	bcyc - 12		ns
T17	Read Data Strobe Low Width (Note 3)	bcyc - 5		ns
T22	Address Valid to $\overline{AS}$ (Note 3)	bch - 18		ns
T23a	Read Data Setup Time to BSCCK	10		ns
T24a	Read Data Hold Time from BSCCK	5		ns
T28	BSCCK to $\overline{MRW}$ (Read) Valid (Note 1)		30	ns
T30	$\overline{DSACK0,1}$ Setup to BSCCK (Note 2)	8		ns
T30a	$\overline{STERM}$ Setup to BSCCK (Note 2)	6		ns
T31	$\overline{DSACK0,1}$ Hold from BSCCK	12		ns
T31a	$\overline{STERM}$ Hold from BSCCK	12		ns

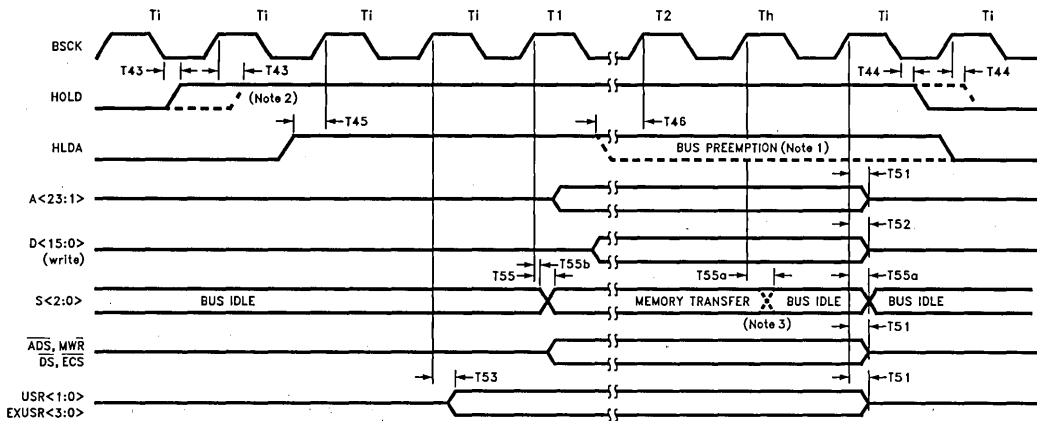
**Note 1:** For successive write operations,  $\overline{MRW}$  remains low.

**Note 2:** Meeting the setup time for  $\overline{DSACK0,1}$  or  $\overline{STERM}$  guarantees that the SONIC-16 will terminate the memory cycle  $1\frac{1}{2}$  bus clocks after  $\overline{DSACK0,1}$  were sampled, or 1 cycle after  $\overline{STERM}$  was sampled. T2 states will be repeated until  $\overline{DSACK0,1}$  or  $\overline{STERM}$  are sampled properly in a low state. If the SONIC-16 samples  $\overline{DSACK0,1}$  or  $\overline{STERM}$  low during the T1 or first T2 state respectively, the SONIC-16 will finish the current access in a total of two bus clocks instead of three (assuming that programmable wait states are set to 0).  $\overline{DSACK0,1}$  are asynchronously sampled and  $\overline{STERM}$  is synchronously sampled.

**Note 3:** bcyc = bus clock cycle time (T3), bch = bus clock high time (T2).

## 7.0 AC and DC Specifications (Continued)

### BUS REQUEST TIMING, BMODE = 0



TL/F/11722-69

Number	Parameter	20 MHz		Units
		Min	Max	
T43	BSCCK to HOLD High (Note 2)		25	ns
T44	BSCCK to HOLD Low (Note 2)		22	ns
T45	HLDA Asynchronous Setup Time to BSCCK	5		ns
T46	HLDA Deassert Setup Time (Note 1)	5		ns
T51	BSCCK to Address, $\overline{ADS}$ , $\overline{MWR}$ , $\overline{DS}$ , $\overline{ECS}$ , USR<1:0> and EXUSR<3:0> TRI-STATE (Note 4)		52	ns
T52	BSCCK to Data TRI-STATE (Note 4)		68	ns
T53	BSCCK to USR<1:0> Valid		50	ns
T55	BSCCK to Bus Status Idle to Non-Idle		40	ns
T55a	BSCCK to Bus Status Non-Idle to Idle (Note 3)		40	ns
T55b	S<2:0> Hold from BSCCK	10		ns

**Note 1:** A block transfer by the SONIC-16 can be pre-empted from the bus by deasserting HLDA provided HLDA is asserted T46 before the rising edge of the last T2 in the current access.

**Note 2:** The assertion edge for HOLD is dependent upon the PH bit in the DCR2. The default situation is shown with a solid line in the timing diagram. T43 and T44 apply for both modes. Also, if HLDA is asserted when the SONIC-16 wants to acquire the bus, HOLD will not be asserted until HLDA has been deasserted first.

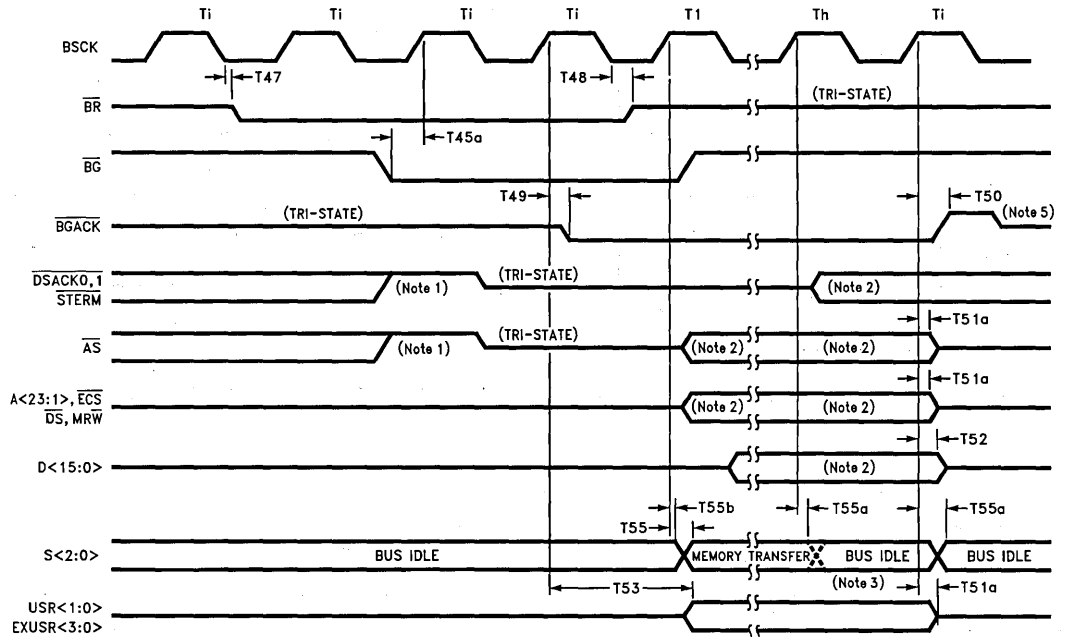
**Note 3:** S<2:0> will indicate IDLE at the end of T2 if the last operation is a read operation, or at the end of Th if the last operation is a write operation.

**Note 4:** This timing value includes an RC delay inherent in the test measurement. These signals typically TRI-STATE 7 ns earlier, enabling other devices to drive these lines without contention.

**Note 5:** For specific timings on these signals (driven by the SONIC-16), see the memory read and memory write timing diagrams on previous pages.

## 7.0 AC and DC Specifications (Continued)

### BUS REQUEST TIMING, BMODE = 1



TL/F/11722-70

Number	Parameter	20 MHz		Units
		Min	Max	
T45a	$\overline{BG}$ Asynchronous Setup Time to BSCCK	8		ns
T47	BSCCK Low to $\overline{BR}$ Low		25	ns
T48	BSCCK Low to $\overline{BR}$ TRI-STATE (Note 4)		30	ns
T49	BSCCK High to $\overline{BGACK}$ Low (Note 1)		30	ns
T50	BSCCK High to $\overline{BGACK}$ High (Note 5)		30	ns
T51a	BSCCK to Address, $\overline{AS}$ , $\overline{MRW}$ , $\overline{DS}$ , $\overline{ECS}$ , $USR<1:0>$ and $EXUSR<3:0>$ TRI-STATE (Note 4)		52	ns
T52	BSCCK to Data TRI-STATE (Note 4)		68	ns
T53	BSCCK to $USR<1:0>$ Valid		50	ns
T55	BSCCK to Bus Status Idle to Non-Idle		40	ns
T55a	BSCCK to Bus Status Non-Idle to Idle (Note 3)		40	ns
T55b	$S<2:0>$ Hold from BSCCK	10		ns

**Note 1:**  $\overline{BGACK}$  is only issued if  $\overline{BG}$  is low and  $\overline{AS}$ ,  $\overline{DSACK0,1}$ ,  $\overline{STERM}$  and  $\overline{BGACK}$  are deasserted.

**Note 2:** For specific timing on these signals driven by the SONIC-16, see the memory read and memory write timing diagrams on previous pages.

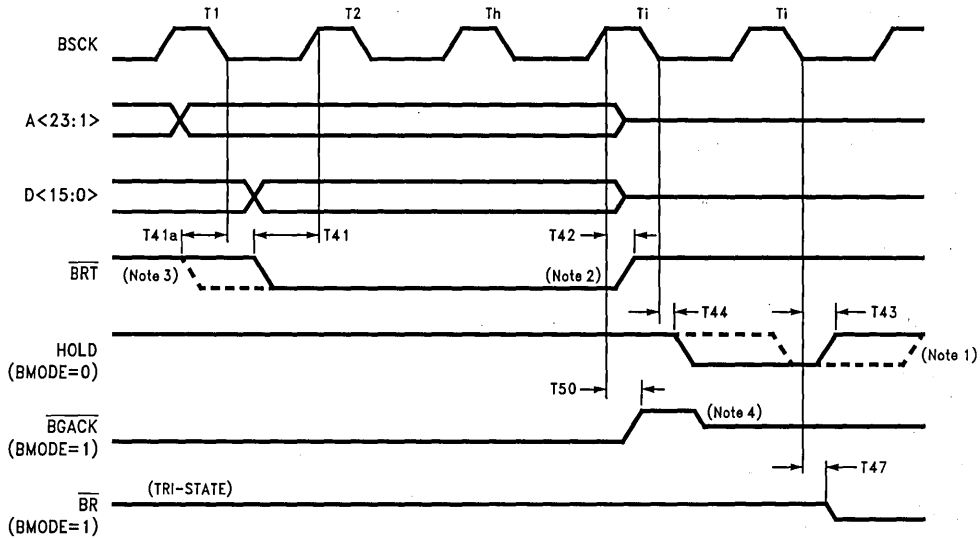
**Note 3:**  $S<2:0>$  will indicate IDLE at the end of  $T_2$  if the last operation is a read operation or at the end of  $T_h$  if the last operation is a write operation.

**Note 4:** This timing value includes an RC delay inherent in our test measurement. These signals typically TRI-STATE 7 ns earlier, enabling other devices to drive these lines without contention.

**Note 5:**  $\overline{BGACK}$  is driven high for approximately  $\frac{1}{2}$  BSCCK before going TRI-STATE.

## 7.0 AC and DC Specifications (Continued)

### BUS RETRY



TL/F/11722-71

Number	Parameter	20 MHz		Units
		Min	Max	
T41	Bus Retry Synchronous Setup Time to BSCCK (Note 3)	5		ns
T41a	Bus Retry Asynchronous Setup Time to BSCCK (Note 3)	5		ns
T42	Bus Retry Hold Time from BSCCK (Note 2)	7		ns
T43	BSCCK to HOLD High (Note 1)		25	ns
T44	BSCCK to HOLD Low (Note 1)		22	ns
T47	BSCCK to BR Low		25	ns
T50	BSCCK to BGACK High (Note 4)		30	ns

**Note 1:** Depending upon the mode, the SONIC-16 will assert and deassert HOLD from the rising or falling edge of BSCCK.

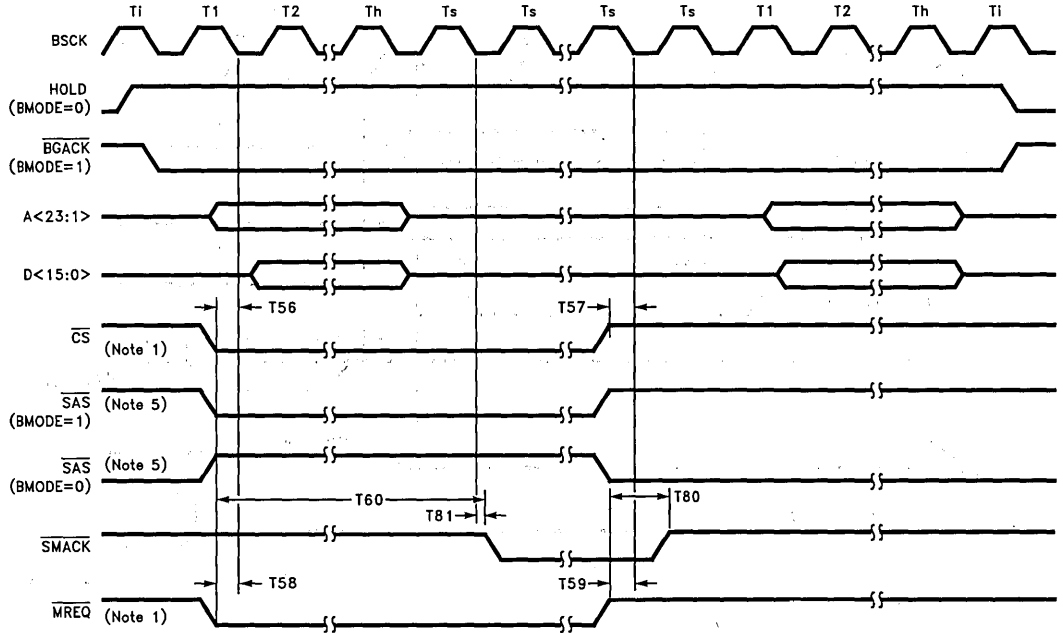
**Note 2:** Unless Latched Bus Retry mode is set (LBR in the Data Configuration Register, Section 4.3.2), BRT must remain asserted until after the Th state. If Latched Bus Retry mode is used, BRT does not need to satisfy T42.

**Note 3:** T41 is for synchronous bus retry and T41a is for asynchronous bus retry (see Section 4.3.2, bit 15, Extended Bus Mode). Since T41a is an asynchronous setup time, it is not necessary to meet it, but doing so will guarantee that the bus exception occurs in the current memory transfer, not the next.

**Note 4:** BGACK is driven high for approximately 1/2 BSCCK before going TRI-STATE.

## 7.0 AC and DC Specifications (Continued)

### MEMORY ARBITRATION/SLAVE ACCESS



TL/F/11722-72

Number	Parameter	20 MHz		Units
		Min	Max	
T56	$\overline{CS}$ Low Asynch. Setup to BSKC (Note 2)	12		ns
T57	$\overline{CS}$ High Asynch. Setup to BSKC	8		ns
T58	$\overline{MREQ}$ Low Asynch. Setup to BSKC (Note 2)	12		ns
T59	$\overline{MREQ}$ High Asynch. Setup to BSKC	12		ns
T60	$\overline{MREQ}$ or $\overline{CS}$ to $\overline{SMACK}$ Low (Notes 3, 4)		1.5 5.5	bcyc
T80	$\overline{MREQ}$ to $\overline{SMACK}$ High		30	ns
T81	BSKC to $\overline{SMACK}$ Low		25	ns

**Note 1:** Both  $\overline{CS}$  and  $\overline{MREQ}$  must not be asserted concurrently. If these signals are successively asserted, there must be at least two bus clocks between the deasserting and asserting edges of these signals.

**Note 2:** It is not necessary to meet the setup times for  $\overline{MREQ}$  or  $\overline{CS}$  since these signals are asynchronously sampled. Meeting the setup time for these signals, however, makes it possible to use T60 to determine exactly when  $\overline{SMACK}$  will be asserted.

**Note 3:** The smaller value for T60 refers to when the SONIC-16 is accessed during an Idle condition and the other value refers to when the SONIC-16 is accessed during non-idle conditions. These values are not tested, but are guaranteed by design. This specification assumes that  $\overline{CS}$  or  $\overline{MREQ}$  is asserted  $\frac{1}{2}$  bus clock before the falling edge that these signals are asynchronously clocked in on (see T56 and T58). If T56 is met for  $\overline{CS}$  or T58 is met for  $\overline{MREQ}$ , then  $\overline{SMACK}$  will be asserted exactly 1 bus clock, when the SONIC-16 was idle, or 5 bus clocks, when the SONIC-16 was in master mode, after the edge that T56 and T58 refer to. (This is assuming that there were no wait states in the current master mode access. Wait states will increase the time for  $\overline{SMACK}$  to go low by the number of wait states in the cycle.)  $\overline{SAS}$  must have been asserted for this timing to be correct. See  $\overline{SAS}$  and  $\overline{CS}$  timing in the Register Read and Register Write timing specifications.

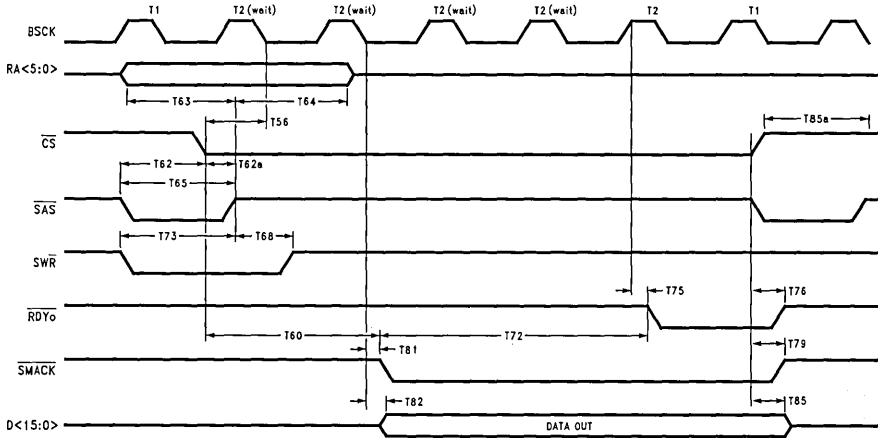
**Note 4:** bcyc = bus clock cycle time (T3).

**Note 5:** The way in which  $\overline{SMACK}$  is asserted is due to  $\overline{CS}$  is not the same as the way in which  $\overline{SMACK}$  is asserted due to  $\overline{MREQ}$ .  $\overline{SMACK}$  goes low as a direct result of the assertion of  $\overline{MREQ}$ , whereas, for  $\overline{CS}$ ,  $\overline{SAS}$  must also be driven low (BMODE = 1) or high (BMODE = 0) before  $\overline{SMACK}$  will be asserted. This means that when  $\overline{SMACK}$  is asserted due to  $\overline{MREQ}$ ,  $\overline{SMACK}$  will remain asserted until  $\overline{MREQ}$  is deasserted. Multiple memory accesses can be made to the shared memory without  $\overline{SMACK}$  ever going high. When  $\overline{SMACK}$  is asserted due to  $\overline{CS}$ , however,  $\overline{SMACK}$  will only remain low as long as  $\overline{SAS}$  is also low (BMODE = 1) or high (BMODE = 0).  $\overline{SMACK}$  will not remain low throughout multiple register accesses to the SONIC-16 because  $\overline{SAS}$  must toggle for each register access. This is an important difference to consider when designing shared memory designs.



## 7.0 AC and DC Specifications (Continued)

### REGISTER READ, BMODE = 0 (Note 1)



TL/F/11722-73

Number	Parameter	20 MHz		Units
		Min	Max	
T56	$\overline{CS}$ Asynch. Setup to BSCCK (Note 4)	12		ns
T60	MREQ or $\overline{CS}$ to $\overline{SMACK}$ Low (Notes 3, 5, 8)		1.5 5.5	bcyc
T62	$\overline{SAS}$ Assertion before $\overline{CS}$ (Note 6)	0		ns
T62a	$\overline{SAS}$ Deassertion after $\overline{CS}$ (Notes 3, 6)		1	bcyc
T63	Register Address Setup to $\overline{SAS}$	10		ns
T64	Register Address Hold Time from $\overline{SAS}$	10		ns
T65	$\overline{SAS}$ Pulse Width (Note 3)	bcyc - 10		ns
T68	$\overline{SWR}$ (Read) Hold from $\overline{SAS}$	8		ns
T72	$\overline{SMACK}$ to $\overline{RDY<0>}$ Low (Notes 3, 8)		2.5	bcyc
T73	$\overline{SWR}$ (Read) Setup to $\overline{SAS}$	0		ns
T75	BSCCK to $\overline{RDY<0>}$ Low		35	ns
T76	$\overline{SAS}$ or $\overline{CS}$ to $\overline{RDY<0>}$ High (Note 2)		30	ns
T79	$\overline{SAS}$ or $\overline{CS}$ to $\overline{SMACK}$ High (Note 2)		30	ns
T81	BSCCK to $\overline{SMACK}$ Low		25	ns
T82	BSCCK to Register Data Valid		83	ns
T85	$\overline{SAS}$ or $\overline{CS}$ to Data TRI-STATE (Notes 2, 7)		60	ns
T85a	Min. $\overline{CS}$ Deassert Time (Note 3)	1		bcyc

**Note 1:** This figure shows a slave access to the SONIC-16 when the SONIC-16 is idle, or rather not in master mode. If the SONIC-16 is a bus master, there will be some differences as noted in the Memory Arbitration/Slave Access diagram. The BSCCK states (T1, T2, etc.) are the equivalent processor states during a slave access.

**Note 2:** If  $\overline{CS}$  is deasserted before the falling edge of  $\overline{SAS}$ , T76, T79 and T85 are referenced from the rising edge of  $\overline{CS}$ .

**Note 3:** bcyc = bus clock cycle time (T3).

**Note 4:** It is not necessary to meet the setup time for  $\overline{CS}$  since this signal is asynchronously sampled. Meeting the setup time for this signal, however, makes it possible to use T60 to determine exactly when  $\overline{SMACK}$  will be asserted.

**Note 5:** The smaller value for T60 refers to when the SONIC-16 is accessed during an Idle condition and the other value refers to when the SONIC-16 is accessed during non-idle conditions. These values are not tested, but are guaranteed by design. This specification assumes that  $\overline{CS}$  is asserted  $\frac{1}{2}$  bus clock before the falling edge that  $\overline{CS}$  is asynchronously clocked in on (see T56). If T56 is met for  $\overline{CS}$ , then  $\overline{SMACK}$  will be asserted exactly 1 bus clock, when the SONIC-16 was idle, or 5 bus clocks, when the SONIC-16 was in master mode, after the edge that T56 refers to. (This is assuming that there were no wait states in the current master mode access. Wait states will increase the time for  $\overline{SMACK}$  to go low by the number of wait states in the cycle.)

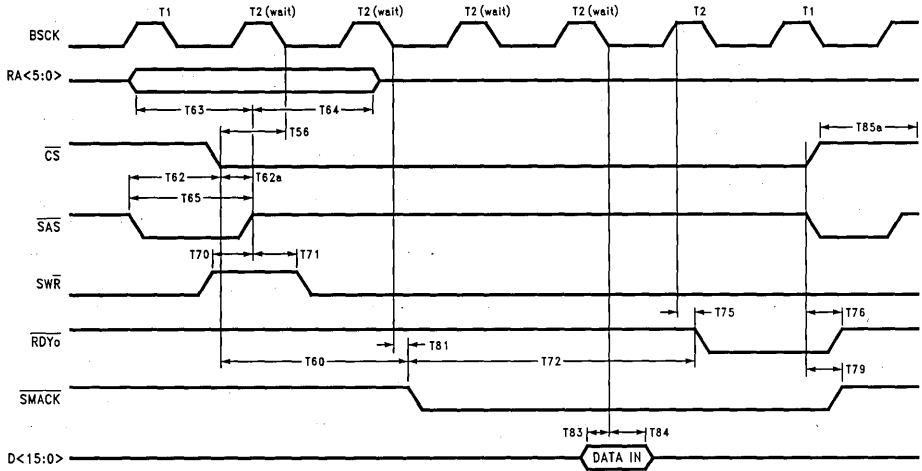
**Note 6:**  $\overline{SAS}$  may be asserted low anytime before or simultaneous to the falling edge of  $\overline{CS}$ . It is suggested that  $\overline{SAS}$  be driven high no later than  $\overline{CS}$ . If necessary, however,  $\overline{SAS}$  may be driven up to 1 BSCCK after  $\overline{CS}$ .

**Note 7:** This timing value includes an RC delay inherent in the test measurement. These signals typically TRI-STATE 7 ns earlier, enabling other devices to drive these lines without contention.

**Note 8:** These values are not tested, but are guaranteed by design. They are provided as a design guideline only.

## 7.0 AC and DC Specifications (Continued)

### REGISTER WRITE, BMODE = 0 (Note 1)



TL/F/11722-74

Number	Parameter	20 MHz		Units
		Min	Max	
T56	$\overline{CS}$ Asynch. Setup to BSCK (Note 4)	12		ns
T60	$\overline{MREQ}$ or $\overline{CS}$ to $\overline{SMACK}$ Low (Notes 3, 5, 7)		1.5 5.5	bcyc
T62	$\overline{SAS}$ Assertion before $\overline{CS}$ (Note 6)	0		ns
T62a	$\overline{SAS}$ Deassertion after $\overline{CS}$ (Notes 3, 6)		1	bcyc
T63	Register Address Setup to $\overline{SAS}$	10		ns
T64	Register Address Hold Time from $\overline{SAS}$	10		ns
T65	$\overline{SAS}$ Pulse Width (Note 3)	bcyc - 10		ns
T70	$\overline{SWR}$ (Write) Setup to $\overline{SAS}$	0		ns
T71	$\overline{SWR}$ (Write) Hold from $\overline{SAS}$	7		ns
T72	$\overline{SMACK}$ to $\overline{RDY<0>}$ Low (Notes 3, 7)	2.5		bcyc
T75	BSCK to $\overline{RDY<0>}$ Low		35	ns
T76	$\overline{SAS}$ or $\overline{CS}$ to $\overline{RDY<0>}$ High (Note 2)		30	ns
T79	$\overline{SAS}$ or $\overline{CS}$ to $\overline{SMACK}$ High (Note 2)		30	ns
T81	BSCK to $\overline{SMACK}$ Low		25	ns
T83	Register Write Data Setup to BSCK	45		ns
T84	Register Write Data Hold from BSCK	20		ns
T85a	Min. $\overline{CS}$ Deassert Time (Note 3)	1		bcyc

**Note 1:** This figure shows a slave access to the SONIC-16 when the SONIC-16 is idle, or rather not in master mode. If the SONIC-16 is a bus master, there will be some differences as noted in the Memory Arbitration/Slave Access diagram. The BSCK states (T1, T2, etc.) are the equivalent processor states during a slave access.

**Note 2:** If  $\overline{CS}$  is deasserted before the falling edge of  $\overline{SAS}$ , T76 and T79 are referenced from the rising edge of  $\overline{CS}$ .

**Note 3:** bcyc = bus clock cycle time (T3).

**Note 4:** It is not necessary to meet the setup time for  $\overline{CS}$  since this signal is asynchronously sampled. Meeting the setup time for this signal, however, makes it possible to use T60 to determine exactly when  $\overline{SMACK}$  will be asserted.

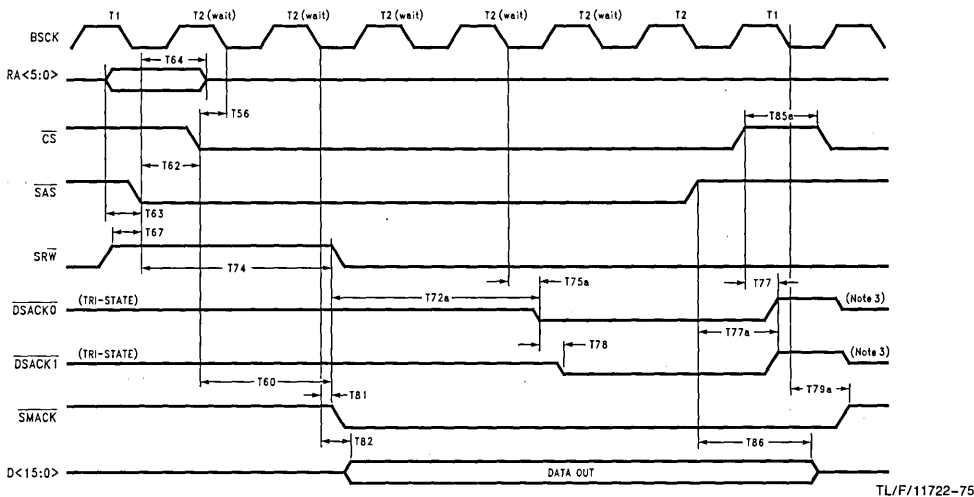
**Note 5:** The smaller value for T60 refers to when the SONIC-16 is accessed during an Idle condition and the other value refers to when the SONIC-16 is accessed during non-idle conditions. These values are not tested, but are guaranteed by design. This specification assumes that  $\overline{CS}$  is asserted  $\frac{1}{2}$  bus clock before the falling edge that  $\overline{CS}$  is asynchronously clocked in on (see T56). If T56 is met for  $\overline{CS}$ , then  $\overline{SMACK}$  will be asserted exactly 1 bus clock, when the SONIC-16 was idle, or 5 bus clocks, when the SONIC-16 was in master mode, after the edge that T56 refers to. (This is assuming that there were no wait states in the current master mode of access. Wait states will increase the time for  $\overline{SMACK}$  to go low by the number of wait states in the cycle.)

**Note 6:**  $\overline{SAS}$  may be asserted low anytime before or simultaneous to the falling edge of  $\overline{CS}$ . It is suggested that  $\overline{SAS}$  be driven high no later than  $\overline{CS}$ . If necessary, however,  $\overline{SAS}$  may be driven up to 1 BSCK after  $\overline{CS}$ .

**Note 7:** These values are not tested, but are guaranteed by design. They are provided as a design guideline only.

## 7.0 AC and DC Specifications (Continued)

### REGISTER READ, BMODE = 1 (Note 1)



TL/F/11722-75

Number	Parameter	20 MHz		Units
		Min	Max	
T56	$\overline{CS}$ Asynch. Setup to BSCCK (Note 5)	12		ns
T60	MREQ or $\overline{CS}$ to $\overline{SMACK}$ Low (Notes 4, 6, 9)		1.5 5.5	bcyc
T62	SAS Assertion before $\overline{CS}$ (Note 7)	0		ns
T63	Register Address Setup to SAS	10		ns
T64	Register Address Hold from $\overline{SAS}$	10		ns
T67	$\overline{SRW}$ (Read) Setup to $\overline{SAS}$	0		ns
T72a	$\overline{SMACK}$ to $\overline{DSACK0,1}$ Low (Notes 4, 9)		2	bcyc
T74	$\overline{SRW}$ (Read) Hold from SAS	50		ns
T75a	BSCCK to $\overline{DSACK0,1}$ Low		35	ns
T77	$\overline{CS}$ to $\overline{DSACK0,1}$ High (Notes 2, 3)		25	ns
T77a	$\overline{SAS}$ to $\overline{DSACK0,1}$ High (Notes 2, 3)		35	ns
T78	Skew between $\overline{DSACK0,1}$		10	ns
T79a	BSCCK to $\overline{SMACK}$ High		30	ns
T81	BSCCK to $\overline{SMACK}$ Low		25	ns
T82	BSCCK to Register Data Valid		83	ns
T85a	Min. $\overline{CS}$ Deassert Time (Note 4)	1		bcyc
T86	$\overline{SAS}$ or $\overline{CS}$ to Register Data TRI-STATE (Notes 2, 8)		60	ns

**Note 1:** This figure shows a slave access to the SONIC-16 when the SONIC-16 is idle, or rather not in master mode. If the SONIC-16 is a bus master, there will be some differences as noted in the Memory Arbitration/Slave Access diagram. The BSCCK states (T1, T2, etc.) are the equivalent processor states during a slave access.

**Note 2:** If  $\overline{CS}$  is deasserted before the rising edge of SAS, T77 and T86 are referenced off the rising edge of  $\overline{CS}$  instead of SAS.

**Note 3:**  $\overline{DSACK0,1}$  are driven high for about 1/2 bus clock before going TRI-STATE.

**Note 4:** bcyc = bus clock cycle time (T3).

**Note 5:** It is not necessary to meet the setup time for  $\overline{CS}$  since this signal is asynchronously sampled. Meeting the setup time for this signal, however, makes it possible to use T60 to determine exactly when  $\overline{SMACK}$  will be asserted.

**Note 6:** The smaller value for T60 refers to when the SONIC-16 is accessed during an Idle condition and the other value refers to when the SONIC-16 is accessed during non-idle conditions. These values are not tested, but are guaranteed by design. This specification assumes that  $\overline{CS}$  is asserted 1/2 bus clock before the falling edge that  $\overline{CS}$  is asynchronously clocked in on (see T56). If T56 is met for  $\overline{CS}$ , then  $\overline{SMACK}$  will be asserted exactly 1 bus clock, when the SONIC-16 was idle, or 5 bus clocks, when the SONIC-16 was in master mode, after the edge that T56 refers to. (This is assuming that there were no wait states in the current master mode access. Wait states will increase the time for  $\overline{SMACK}$  to go low by the number of wait states in the cycle.)

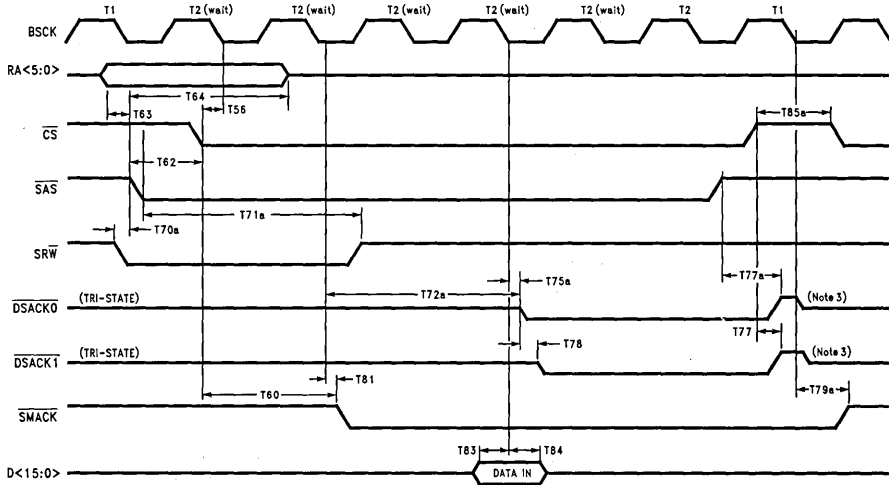
**Note 7:** SAS may be asserted at anytime before or simultaneous to the falling edge of  $\overline{CS}$ .

**Note 8:** This timing value includes an RC delay inherent in the test measurement. These signals typically TRI-STATE 7 ns earlier, enabling other devices to drive these lines without contention.

**Note 9:** These values are not tested, but are guaranteed by design. They are provided as in design guideline only.

## 7.0 AC and DC Specifications (Continued)

### REGISTER WRITE, BMODE = 1 (Note 1)



TL/F/11722-76

Number	Parameter	20 MHz		Units
		Min	Max	
T56	$\overline{CS}$ Asynch. Setup to BCLK (Note 5)	12		ns
T60	$\overline{MREQ}$ or $\overline{CS}$ to $\overline{SMACK}$ Low (Notes 4, 6, 8)		1.5 5.5	bcyc
T62	$\overline{SAS}$ Assertion before $\overline{CS}$ (Note 7)	0		ns
T63	Register Address Setup to $\overline{SAS}$	10		ns
T66	Register Address Hold from $\overline{SAS}$	10		ns
T70a	$\overline{SRW}$ (Write) Setup to $\overline{SAS}$	0		ns
T71a	$\overline{SRW}$ (Write) Hold from $\overline{SAS}$	10		ns
T72a	$\overline{SMACK}$ to $\overline{DSACK0,1}$ Low (Notes 4, 8)		2	bcyc
T75b	BCLK to $\overline{DSACK0,1}$ Low		44	ns
T77	$\overline{CS}$ to $\overline{DSACK0,1}$ High (Notes 2, 3)		25	ns
T77a	$\overline{SAS}$ to $\overline{DSACK0,1}$ High (Notes 2, 3)		35	ns
T78	Skew between $\overline{DSACK0,1}$		10	ns
T79a	BCLK to $\overline{SMACK}$ High		30	ns
T81	BCLK to $\overline{SMACK}$ Low		25	ns
T83	Register Write Data Setup to BCLK	45		ns
T84	Register Write Data Hold from BCLK	20		ns
T85a	Min. $\overline{CS}$ Deassert Time (Note 4)	1		bcyc

**Note 1:** This figure shows a slave access to the SONIC-16 when the SONIC-16 is idle, or rather not in master mode. If the SONIC-16 is a bus master, there will be some differences as noted in the Memory Arbitration/Slave Access diagram. The BCLK states (T1, T2, etc.) are the equivalent processor states during a slave access.

**Note 2:** If  $\overline{CS}$  is deasserted before the rising edge of  $\overline{SAS}$ , then T77 is referenced off the rising edge of  $\overline{CS}$  instead of  $\overline{SAS}$ .

**Note 3:**  $\overline{DSACK0,1}$  are driven high for about 1/2 bus clock before going TRI-STATE.

**Note 4:** bcyc = bus clock cycle time (T3).

**Note 5:** It is not necessary to meet the setup time for  $\overline{CS}$  since this signal is asynchronously sampled. Meeting the setup time for this signal, however, makes it possible to use T60 to determine exactly when  $\overline{SMACK}$  will be asserted.

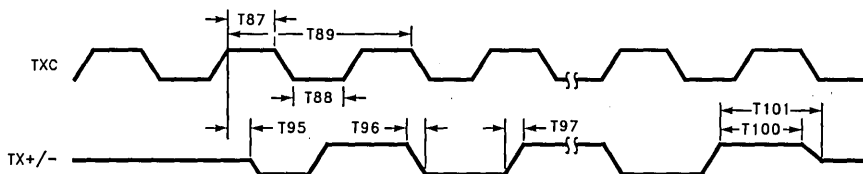
**Note 6:** The smaller value for T60 refers to when the SONIC-16 is accessed during an Idle condition and the other value refers to when the SONIC-16 is accessed during non-idle conditions. These values are not tested, but are guaranteed by design. This specification assumes that  $\overline{CS}$  is asserted 1/2 bus clock before the falling edge that  $\overline{CS}$  is asynchronously clocked in on (see T56). If T56 is met for  $\overline{CS}$ , then  $\overline{SMACK}$  will be asserted exactly 1 bus clock, when the SONIC-16 was idle, or 5 bus clocks, when the SONIC-16 was in master mode, after the edge that T56 refers to. (This is assuming that there were no wait states in the current master mode access. Wait states will increase the time for  $\overline{SMACK}$  to go low by the number of wait states in the cycle.)

**Note 7:**  $\overline{SAS}$  may be asserted low anytime before or simultaneous to the falling edge of  $\overline{CS}$ .

**Note 8:** These values are not tested, but are guaranteed by design. They are provided as a design guideline only.

## 7.0 AC and DC Specifications (Continued)

### ENDEC TRANSMIT TIMING (INTERNAL ENDEC MODE)



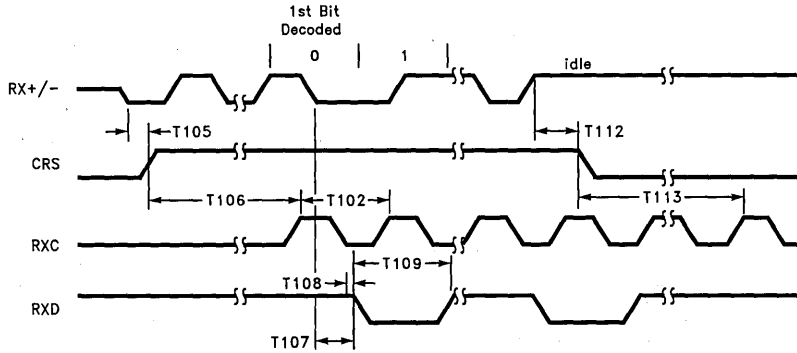
TL/F/11722-77

Number	Parameter	Min	Max	Units
T87	Transmit Clock High Time (Note 1)	40		ns
T88	Transmit Clock Low Time (Note 1)	40		ns
T89	Transmit Clock Cycle Time (Note 1)	99.99	100.01	ns
T95	Transmit Output Delay (Note 1)		55	ns
T96	Transmit Output Fall Time (80% to 20%, Note 1)		7	ns
T97	Transmit Output Rise Time (20% to 80%, Note 1)		7	ns
T98	Transmit Output Jitter (Not Shown)	0.5 Typ		ns
T100	Transmit Output High before Idle (Half Step)	200		ns
T101	Transmit Output Idle Time (Half Step)		8000	ns

**Note 1:** This specification is provided for information only and is not tested.

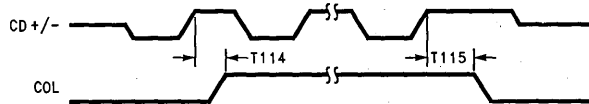
## 7.0 AC and DC Specifications (Continued)

### ENDEC RECEIVE TIMING (INTERNAL ENDEC MODE)



TL/F/11722-78

### ENDEC COLLISION TIMING



TL/F/11722-79

Number	Parameter	Min	Max	Units
T102	Receive Clock Duty Cycle Time (Note 1)	40	60	ns
T105	Carrier Sense on Time		70	ns
T106	Data Acquisition Time		700	ns
T107	Receive Data Output Delay		150	ns
T108	Receive Data Valid from RXC		10	ns
T109	Receive Data Stable Valid Time	90		ns
T112	Carrier Sense Off Delay (Note 2)		155	ns
T113	Minimum Number of RXCs after CRS Low	5		rcyc (Note 3)
T114	Collision Turn On Time		55	ns
T115	Collision Turn Off Time		250	ns

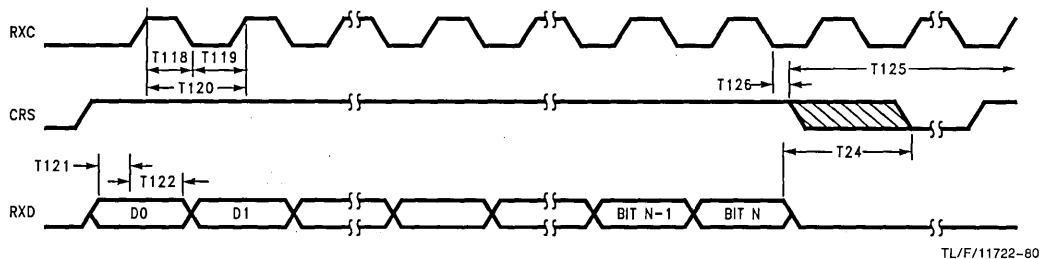
**Note 1:** This parameter is measured at the 50% point of each clock edge.

**Note 2:** When CRSi goes low, it remains low for a minimum of 2 receive clocks (RXC).

**Note 3:** rcyc = receive clocks.

## 7.0 AC and DC Specifications (Continued)

### ENDEC-MAC SERIAL TIMING FOR RECEPTION (EXTERNAL ENDEC MODE)

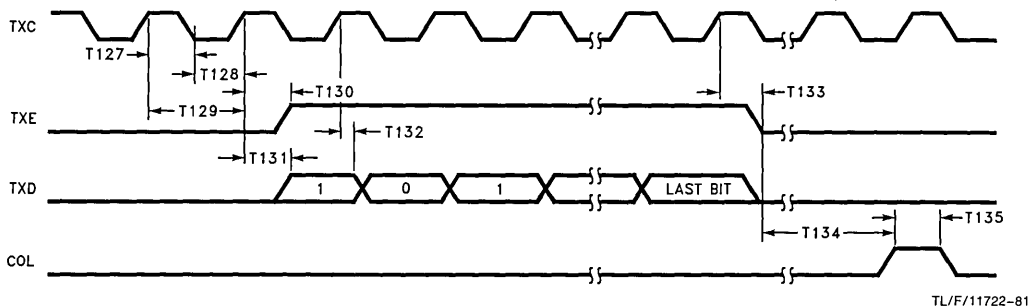


Number	Parameter	Min	Max	Units
T118	Receive Clock High Time	35		ns
T119	Receive Clock Low Time	35		ns
T120	Receive Clock Cycle Time	90	110	ns
T121	RXD Setup to RXC	20		ns
T122	RXD Hold from RXC	15		ns
T124	Maximum Allowed Dribble Bits		6	Bits
T125	Receive Recovery Time (Note 2)			
T126	RXC to Carrier Sense Low (Note 1)		1	rcyc

**Note 1:** tcyc = transmit clocks, rcyc = receive clocks, bcyc = T3.

**Note 2:** This parameter refers to longest time (not including wait-states) the SONIC-16 requires to perform its end of receive processing and be ready for the next start of frame delimiter. This time is 4 tcyc + 36 bcyc. This is guaranteed by design and is not tested.

### ENDEC-MAC SERIAL TIMING FOR TRANSMIT (NO COLLISION)

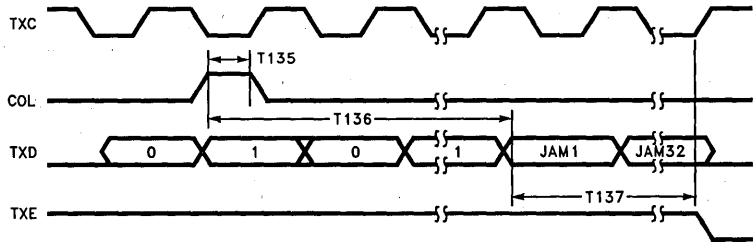


Number	Parameter	Min	Max	Units
T127	Transmit Clock High Time	40		ns
T128	Transmit Clock Low Time	40		ns
T129	Transmit Clock Cycle Time	90	110	ns
T130	TXC to TXE High		40	ns
T131	TXC to TXD Valid		15	ns
T132	TXD Hold Time from TXC	5		ns
T133	TXC to TXE Low		40	ns
T134	TXE Low to Start of CD Heartbeat (Note 1)		64	tcyc
T135	Collision Detect Width (Note 1)	2		tcyc

**Note 1:** tcyc = transmit clock.

## 7.0 AC and DC Specifications (Continued)

### ENDEC-MAC SERIAL TIMING FOR TRANSMISSION (COLLISION)



TL/F/11722-82

Number	Parameter	Min	Max	Units
T135	Collision Detect Width (Note 1)	2		tcyc
T136	Delay from Collision		8	tcyc
T137	Jam Period		32	tcyc

Note 1: tcyc = transmit clock.

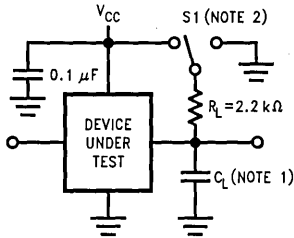


## 8.0 AC Timing Test Conditions

All specifications are valid only if the mandatory isolation is employed and all differential signals are taken to be at the AUI side of the pulse transformer.

Input Pulse Levels (TTL/CMOS)	GND to 3.0V
Input Rise and Fall Times (TTL/CMOS)	5 ns
Input and Output Reference Levels (TTL/CMOS)	1.5V
Input Pulse Levels (Diff.)	-350 mV to -1315 mV
Input and Output Reference Levels (Diff.)	50% Point of the Differential
TRI-STATE Reference Levels	Float ( $\Delta V$ ) $\pm 0.5V$

### OUTPUT LOAD (See Figure below)



TL/F/11722-83

**Note 1:** 50 pF, includes scope and jig capacitance.

**Note 2:** S1 = Open for timing tests for push pull outputs.

S1 =  $V_{CC}$  for  $V_{OL}$  test.

S1 = GND for  $V_{OH}$  test.

S1 =  $V_{CC}$  for High Impedance to active low and active low to High Impedance measurements.

S1 = GND for High Impedance to active high and active High to High Impedance measurements.

### PIN CAPACITANCE

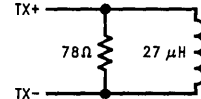
$T_A = 25^\circ C$ ,  $f = 1$  MHz

Symbol	Parameter	Typ	Units
$C_{IN}$	Input Capacitance	7	pF
$C_{OUT}$	Output Capacitance	7	pF

### DERATING FACTOR

Output timing is measured with a purely capacitive load of 50 pF. The following correction factor can be used for other loads:  $C_L \geq 50$  pF + 0.05 ns/pF.

### AUI Transmit Test Load



TL/F/11722-84

**Note:** In the above diagram, the TX+ and TX- signals are taken from the AUI side of the isolation (pulse transformer). The pulse transformer used for all testing is a selected 100  $\mu H \pm 0.1\%$  Pulse Engineering PE64103.

# 32-Bit Bus Master Ethernet Interface for the 68030 (Using the Macintosh SE/30)

National Semiconductor  
Application Note 691  
William Harmon



## OVERVIEW

National Semiconductor's SE/30 Ethernet adapter provides a high performance, 32-bit, bus master network connection for Apple's 68030 based compact Macintosh computer. This design is based around National Semiconductor's Systems Oriented Network Interface Controller (SONIC™, DP83932), which interfaces directly to the extension slot of the SE/30. The SE/30 design also serves as a model for designing the SONIC onto the mother board of a 68030 based system, since the SE/30's one expansion slot is essentially a direct connection to the Motorola 68030.

A block diagram of the adapter can be seen in *Figure 1*. The SE/30 Ethernet adapter operates synchronously with the 16 MHz SE/30 mother board and accesses the necessary transmit and receive buffers directly in the system's main memory, via 16 MHz 3 cycle asynchronous DMA operations. At this rate, the bus utilization for the buffering of a single packet is approximately 6% of the total bus bandwidth.

In addition to it's high performance DMA, the SONIC also has an on board Ethernet Manchester Encoder/Decoder (ENDEC), which allows the SONIC to communicate directly

with any AUI interface. In fact, the SE/30 board has the capability to be connected to a network through either thin wire (10Base2) or AUI drop cable (10Base5) Ethernet.

It is also worth mentioning that the SE/30 adapter supports the use of Macintosh Nubus Slot Manager features, such as interrupt handling, with an on board Slot Manager PROM. This does not cause the board to incur any extra cost, since some type of PROM must already be used to store the adapter's Ethernet address.

## FEATURES

- 32-bit bus master system interface
- Asynchronous high speed 3 cycle DMA
- 100% on card address filtering, via the SONIC's on board Content Addressable Memory (CAM)
- Minimal number of components
- Supports both AUI cable and thin wire Ethernet
- Optimal placement of receive and transmit data and descriptors in system memory
- Supports Macintosh Slot Manager
- Portable to 68030 mother board designs

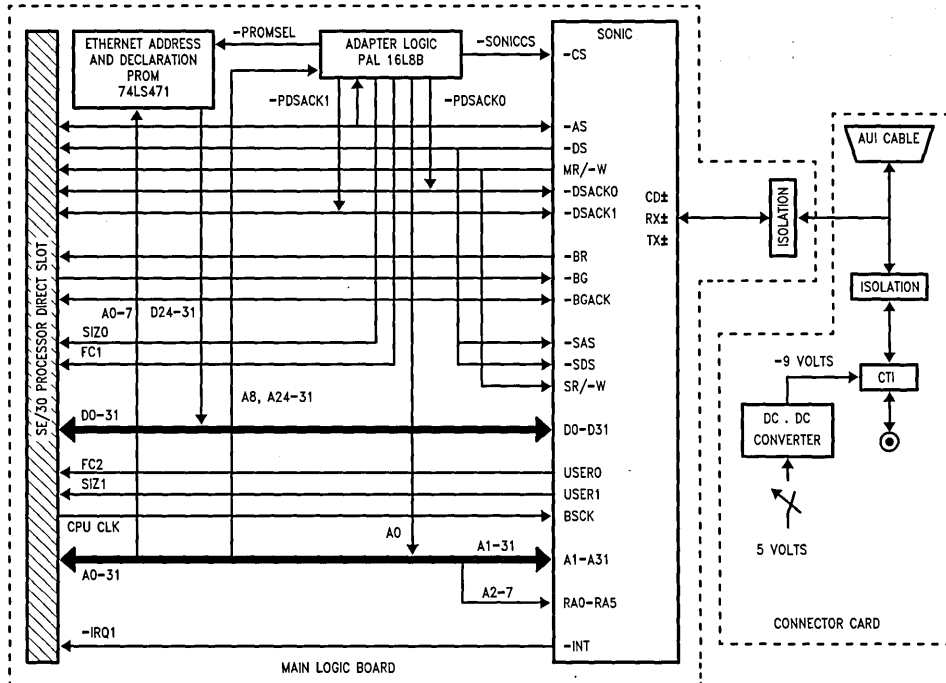


FIGURE 1. Adapter Block Diagram

TL/F/10848-1

## FUNCTIONAL OVERVIEW

### System Interface

The Macintosh SE/30 provides a single 32-bit expansion slot, which basically consists of the control, data, and address signals of the mother board's CPU, the 68030. In addition to these signals, the expansion slot also provides Nubus compatible interrupt lines, so that Slot Manager software can be incorporated, if a Slot Manager PROM is placed on the card. Hence, the hardware interface between the SE/30 and the Ethernet adapter is essentially a 68030 bus interface, while the driver software interface is similar to that of a Macintosh II Nubus adapter. This solution is optimally achieved through the use of National Semiconductor's SONIC, whose 32-bit data and address buses and control signals interface directly to those of the Motorola 68030.

The SE/30 Ethernet adapter operates in both a slave and master mode. When operating in a bus master mode, the SE/30 Ethernet adapter arbitrates with the host system for control of the SE/30 bus and proceeds to operate as a 32-bit DMA engine between the system memory and the network. A block diagram of this interface can be found in *Figure 1*. The bus master mode of operation allows for the use of system memory, instead of on card RAM, for the buffering of transmit and receive data and their descriptors. Master operation is facilitated by the SONIC, which is at the heart of this adapter's design. The SONIC provides the complete implementation of the IEEE 802.3 specification from the AUI interface through the MAC layer, as well as performs a direct system interface to the 68030. In fact, when interfacing to the SE/30 backplane, the SONIC carries out 16 MHz 3 cycle asynchronous DMA, which is fully synchronous with the 16 MHz mother board of the SE/30. This enables the SE/30 adapter to operate on the bus in the same fashion as the 68030 and utilize only 6% of the bus bandwidth, during an Ethernet reception or transmission. The bus master design provides for the highest possible throughput between the system and the network, while at the same time requiring only a minimum of parts to implement.

When the adapter is a slave, the host system accesses either the Slot Manager PROM or the SONIC's registers. All slave operations are done via memory reads and writes, since both the PROM and the SONIC registers are mapped into system memory. The slave architecture is depicted in the adapter block diagram (*Figure 1*). While in the slave mode, the SONIC once again provides a direct interface to the SE/30. The only necessary interface logic is the address decode for the SONIC chip select (-SONICCS). At this point, it is worth noting that the slave address strobe (-SAS) of the SONIC is connected to the data strobe (-DS) of the SE/30 instead of the SE/30's address strobe (-AS). This is due to the operation of the SE/30 backplane and will be further discussed in the design section of this document. However, it is important to remember that in interfacing directly to a 68030 CPU the SONIC's -SAS would be connected directly to the 68030's -AS.

### Network Interface

With respect to the adapter's physical layer design, both AUI drop cable Ethernet and thin wire Ethernet are supported. The SE/30 adapter consists of two boards, the main logic board, which contains the SONIC, and the connector card which provides for the AUI and thin wire network connections. The connector card, whose block diagram is shown in *Figure 2*, contains a 15-pin AUI drop cable connector for standard drop cable Ethernet implementations, as well as a thin wire Ethernet connection via the National Semiconductor coaxial transceiver interface (CTI, DP8392).

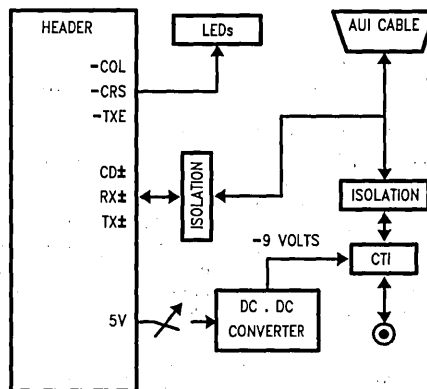


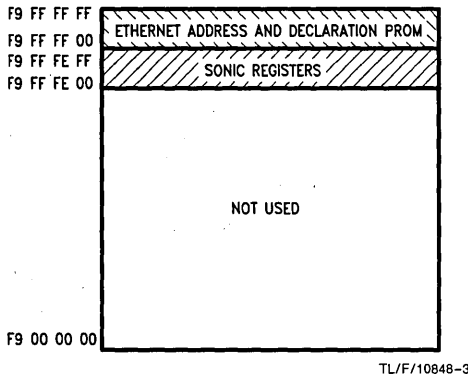
FIGURE 2. Connector Card Block Diagram

Either of these network connections can be chosen through the use of a single jumper. In either case, the AUI signals (RX $\pm$ , TX $\pm$ , and CD $\pm$ ) are sent back to the main logic board, where the SONIC resides. These signals are interfaced to the ENDEC portion of the SONIC, which provides for communication between the AUI interface and the non-return to zero (NRZ) signals (RXD, TXD, and COL) of the Media Access Control (MAC) module of the SONIC. It should be noted that the integrated ENDEC module of the SONIC alleviates the need for an external Ethernet Manchester encoder/docoder, such as National's CMOS Serial Network Interface (CMOS SNI, DP83910).

### BOARD ARCHITECTURE AND DESIGN

#### Memory Map

As stated previously, the SE/30 adapter is completely mapped into the addressable memory space of the SE/30. A diagram of the memory map can be found in *Figure 3*. The board is mapped into the memory locations F9FFFFFF through F9000000. Locations F9FFFFFF through F9FFFF00 contain the Ethernet address and declaration PROM. This region contains the adapter's Ethernet address as well as the declaration data that is necessary for the adapter's interrupt service routine to take advantage of the Slot Manager features, which are provided by the SE/30 operating system.



**FIGURE 3. SE/30 Adapter's Memory Map**

The SE/30 specification states that this declaration data begin at memory location F9FFFFFF, if the adapter's interrupt is generated on the system's IRQ 1 interrupt line. This is the case with the SE/30 adapter. The six byte Ethernet address immediately follows the Slot Manager software in the PROM. It should be noted that the data in the PROM is all byte addressable. In addition to the declaration information, the SONIC registers are also mapped into memory. These registers are mapped into locations F9FFFE0 through F9FFFEFF. The SONIC's registers are mapped as 32-bit addressable quantities, in spite of the fact that internally all SONIC registers are only 16 bits wide. This is due to the fact that the SONIC will always respond as a 32-bit port, since it is programmed to operate in 32-bit mode.

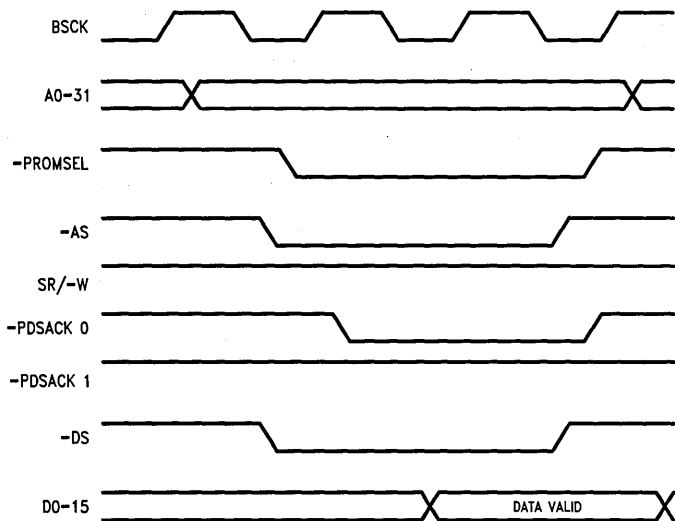
**Slave Operation**

When operating as a slave, the Ethernet adapter appears as a block of memory to the host system. In slave mode, either the SONIC or PROM can be accessed. Timing diagrams for slave accesses appear in Figures 4-6. In the case of accessing the PROM (Figure 4), the 68030 will issue a byte read command. The adapter logic will decode address lines

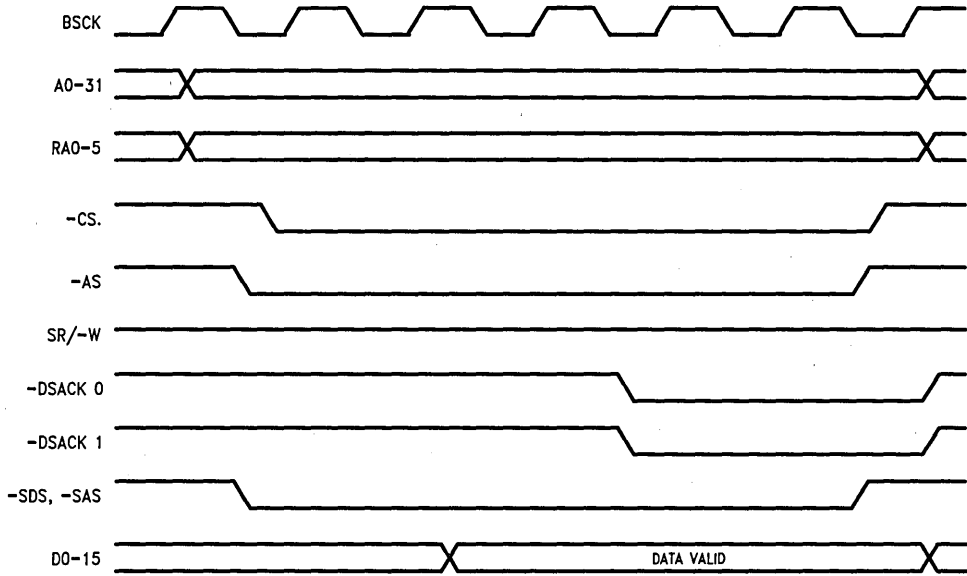
8 and 24 through 31 and recognize the fact that the SE/30 adapter's PROM is being selected. Once the 68030 asserts its address strobe, the logic will issue an enable signal to the PROM (-PROMSEL) and assert the cycle acknowledge signals (-PDSACK0 and -PDSACK1) back to the 68030, in order to indicate the adapter's acknowledgement of a byte access. In parallel with the logic's operation, the PROM will decode the address it is being given (A0-A7) and begin to source data after receiving the -PROMSEL signal. Finally, the 68030 will then finish the read cycle, at which point the adapter logic will then deassert -PROMSEL, -PDSACK0, and -PDSACK1.

As seen in Figures 5 and 6, slave accesses to the SONIC are completely compatible with the bus of the 68030 processor. The only deviation is the connection of -SAS to the 68030's -DS instead of -AS. This is due to the fact that during slave writes a glitch may occur on the memory read/write line (MR/-W) of the SE/30 backplane, while -AS is being asserted. This is fatal, since the SONIC latches the value of the slave read/write line (SR/-W, which is connected to MR/-W) with the falling edge of -SAS. The connection of the 68030's -DS to -SAS solves this problem. It should also be noted that the SONIC is mapped into memory as a 32-bit peripheral and will respond accordingly. However, only the lower 16 data lines (D0-D15) will be valid inputs and outputs during slave accesses.

A 32-bit mapping was selected, since the SONIC is programmed to operate in 32-bit mode, which causes the SONIC to respond with the acknowledge signals of a 32-bit port (-DSACK0 = 0 and -DSACK1 = 0). The only adapter logic necessary to facilitate this interface is the decode of address lines 8 and 24 through 31, along with the address strobe (-AS), to generate a chip select signal to the SONIC (-SONICCS). When accessing the SONIC registers, the 68030 will perform either a 32-bit read or a 32-bit write. Once -SONICCS is asserted the SONIC will respond with the acknowledge signals (-DSACK0 and -DSACK1) and appropriately source or sink data. The deassertion of -DS by the 68030 signals the end of the cycle and causes the SONIC to deassert -DSACK0 and -DSACK1 and terminate the slave cycle.

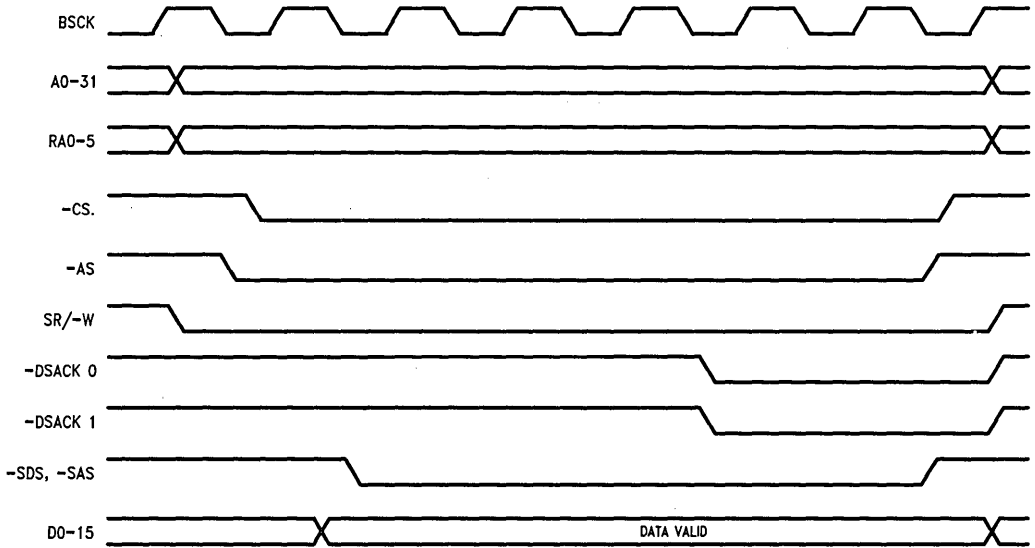


**FIGURE 4. PROM Read**



TL/F/10848-5

FIGURE 5. SONIC Slave Read



TL/F/10848-6

FIGURE 6. SONIC Slave Write

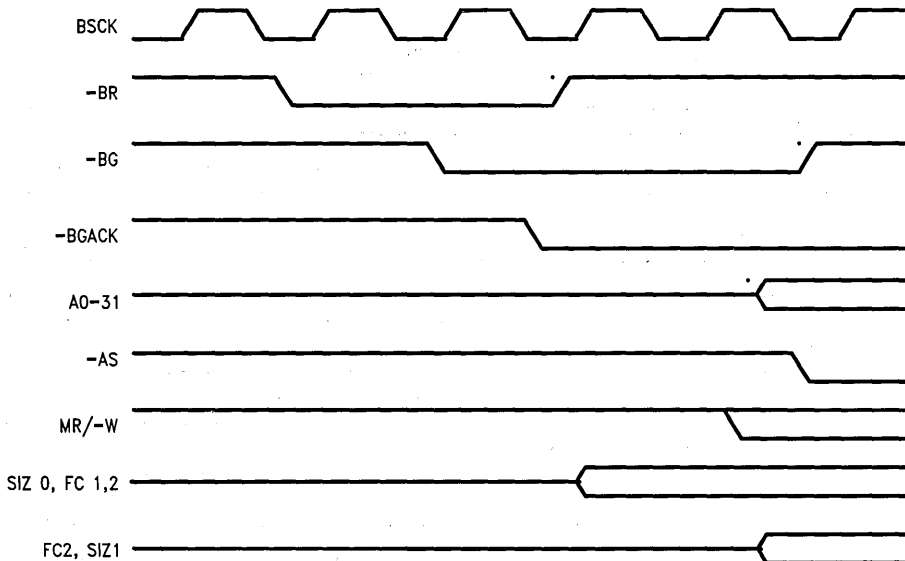
### Master Operation

As stated previously, the SE/30 Ethernet adapter contains no local memory. All Ethernet data and descriptors are stored in system memory, which is accessed directly by the adapter. More specifically, the system memory is accessed directly by the SONIC, which interfaces directly to the 68030 mother board bus of the SE/30. When a master DMA access is required, the SE/30 adapter will arbitrate for the SE/30 system bus. This arbitration is performed by the SONIC, which connects directly to the bus arbitration signals of the 68030. This is depicted in the adapter block diagram (Figure 1). A timing diagram of the arbitration cycle can be found in Figure 7. When the SONIC initiates a request for the system bus, it asserts the bus request signal (-BR) and waits for the bus grant signal (-BG) to be returned by the system. Once the bus grant signal is received, the SONIC will take ownership of the bus by asserting the open collector bus grant acknowledge signal (-BGACK), when the host system's -AS, -DSACK0, -DSACK1, and -BGACK are all deasserted. Once -BGACK is asserted, the SONIC removes the bus request signal.

After acquiring the bus the SE/30 adapter will begin to perform 16 MHz 3 cycle asynchronous DMA on the system bus. This function is also facilitated by the SONIC, whose direct 68030 interface allows the SE/30 adapter to operate on this bus with virtually no interface logic. As seen in Figure 7, the bus interface signals on the SONIC are attached directly to those of the SE/30 backplane.

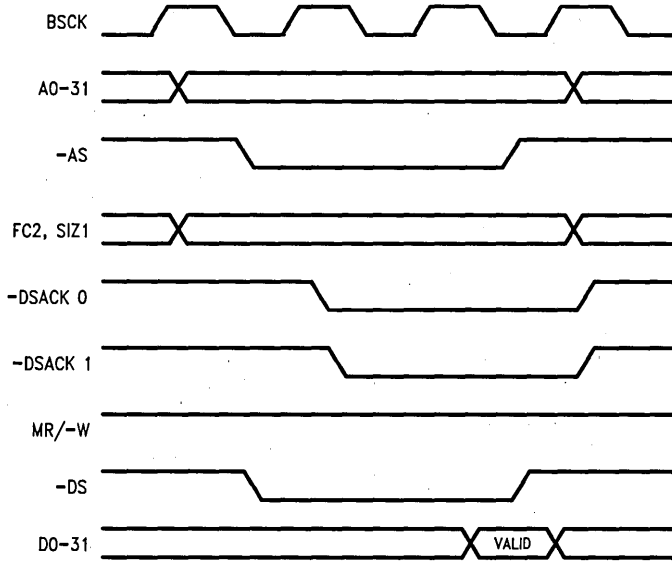
Timing diagrams of the adapter's master read and master write cycles appear in Figures 8 and 9. The only external interface logic required is for the generation of Function Code bit 1 (FC1), SIZ0, and address line 0 (A0). These lines are not provided directly by the SONIC, but are formulated in the adapter logic. All three signals are driven low upon the SONIC's assertion of -BGACK. It should also be noted that Function Code bits 0 and 2 (FC0 and FC2) and the SIZ1 signal are also not provided directly by the SONIC. However, these signals require no extra logic. The FC0 signal is tied high through a backplane resistor and requires no board connection, since the SONIC should only access memory areas which correspond to function codes with the least significant bit set high. These areas are user data space (FC2, FC1, FC0 = 001) and supervisor data space (FC2, FC1, FC0 = 101).

The FC2 and SIZ1 signals are not defined on the SONIC, but they can be generated by using the user 0 and user 1 pins. The user 0 and 1 outputs can be programmed by the programmable output bits (PO0 and PO1) in the SONIC's data configuration register (DCR). The output timing for these signals corresponds to the timing for the SONIC's address lines, which is the correct timing for both FC2 and SIZ1. By programming PO0 with a 0 or 1, the adapter can be made to access either the user data space or supervisor data space of the SE/30's system memory. In order to provide the correct SIZ1 signal for 32-bit operation the PO1 bit should be programmed to a 0.



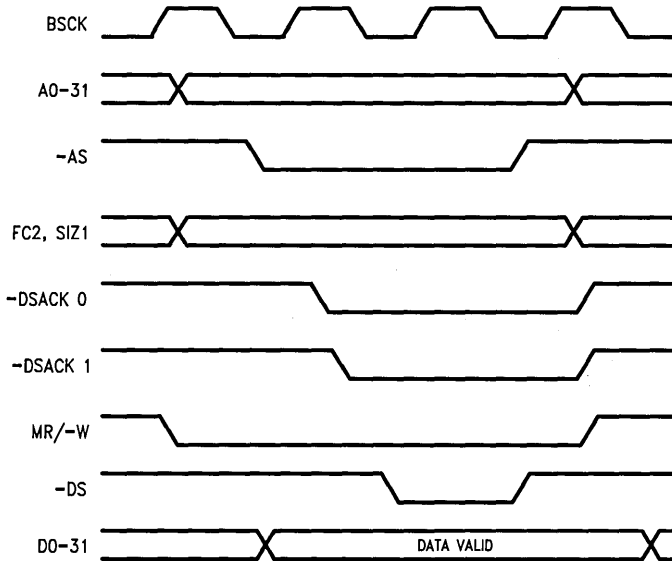
TL/F/10848-7

FIGURE 7. SONIC Master Arbitration



TL/F/10848-8

FIGURE 8. Master Read



TL/F/10848-9

FIGURE 9. Master Write

## Physical Layer

The physical layer interface for the SE/30 adapter resides on the connector card, which attaches to the back of the SE/30. The connector board is linked to the main logic board through a ribbon cable that attaches to a 20-pin header on the main logic board (J2) and another 20-pin header on the connector board (J3). These two headers can be seen on the adapter schematics, which appear at the end of the manual of this application note. The adapter can be used in either a thin wire or standard drop cable Ethernet environment. When the adapter is used in a thin wire Ethernet application, jumper 3 (JB3) must have the jumper covering both posts. This enables the DC-to-DC converter to receive a 5V input from the SE/30 backplane and convert this to a -9V output, which is required by National Semiconductor's Coaxial Transceiver Interface (CTI, DP8392). The CTI provides an interface between the 10 MHz Manchester encoded coax cable and the 10 MHz Manchester encoded differential signals of the SONIC's ENDEC. In the case of a standard drop cable Ethernet application, JB3 is left uncovered so that the CTI will not receive power. This allows the signals of the SONIC's ENDEC to pass directly to the AUI cable, via the 15-pin AUI connector. In examining the schematic of the physical layer design, it can be seen that there is a pulse transformer at the AUI side of the CTI. This is placed here to isolate the CTI from the SONIC's ENDEC signals, when the AUI drop cable connection is being employed. This transformer also provides the IEEE 802.3 specified isolation between the coax and the differential AUI signals, when thin wire Ethernet is being used. It is also

necessary to provide a termination for the 78Ω AUI cable's differential receive and collision pair (RX± and CD±). This is the reason for the 39Ω -1% resistors and 0.01 μF capacitors that are shown in the schematic.

Since the ENDEC resides within the SONIC, two components of the physical layer design are located on the main logic board, which can be seen on the schematics. First, each one of the transmit pairs (TX+ and TX-) requires a 270Ω non-precision pull down resistor (R1 and R2) to complete the internal source follower amplifiers that drive these signals. Second, there is an isolation transformer (T1) placed between the differential signals of the SONIC's ENDEC and the header for the ribbon cable. This isolation is necessary to guarantee that the SONIC meets the IEEE 802.3 fail safe specification of a 16V DC level appearing on the AUI cable's differential signals. The external isolation is necessary, due to the fact that in the powered down state the CMOS process, in which the SONIC is manufactured, may not be able to withstand this voltage.

The final feature of the physical layer design is the diagnostic LEDs. The yellow LED indicates that the ENDEC carrier sense signal (CRS) is asserted. An inverted version of CRS drives this LED. The green LED indicates that a transmission is in progress, and the red LED indicates the presence of a collision. The transmission LED and collision LED are driven by inversions of the SONIC's transmit enable (TXE) and collision output (COL) signals, respectively. The signals for the LEDs are supplied from the main logic board, via the ribbon cable that connects the two boards.

## ADAPTER LOGIC EQUATIONS

The following is the set of logic equations that are necessary to implement the adapter logic block found in *Figure 1*. As shown in the schematics, this logic can be implemented in a single 16L8B PAL.

### Inputs

A31, A30, A29	Pin	1, 2, 3
A28, A27, A26	Pin	4, 5, 6
A25, A24, A8	Pin	7, 8, 9
AS, BGACK	Pin	11, 13

### Outputs

A0	Pin	12; Address line 0 (TRI-STATE)
SIZ0	Pin	14; 68030 SIZ0 signal (TRI-STATE)
FC1	Pin	15; 68030 Function Code 1 signal (TRI-STATE)
PDSACK1	Pin	16; PROM cycle acknowledgement (TRI-STATE)
PDSACK0	Pin	17; PROM cycle acknowledgement (TRI-STATE)
-PROMSEL	Pin	18; PROM chip select
-SONICCS	Pin	19; SONIC chip select

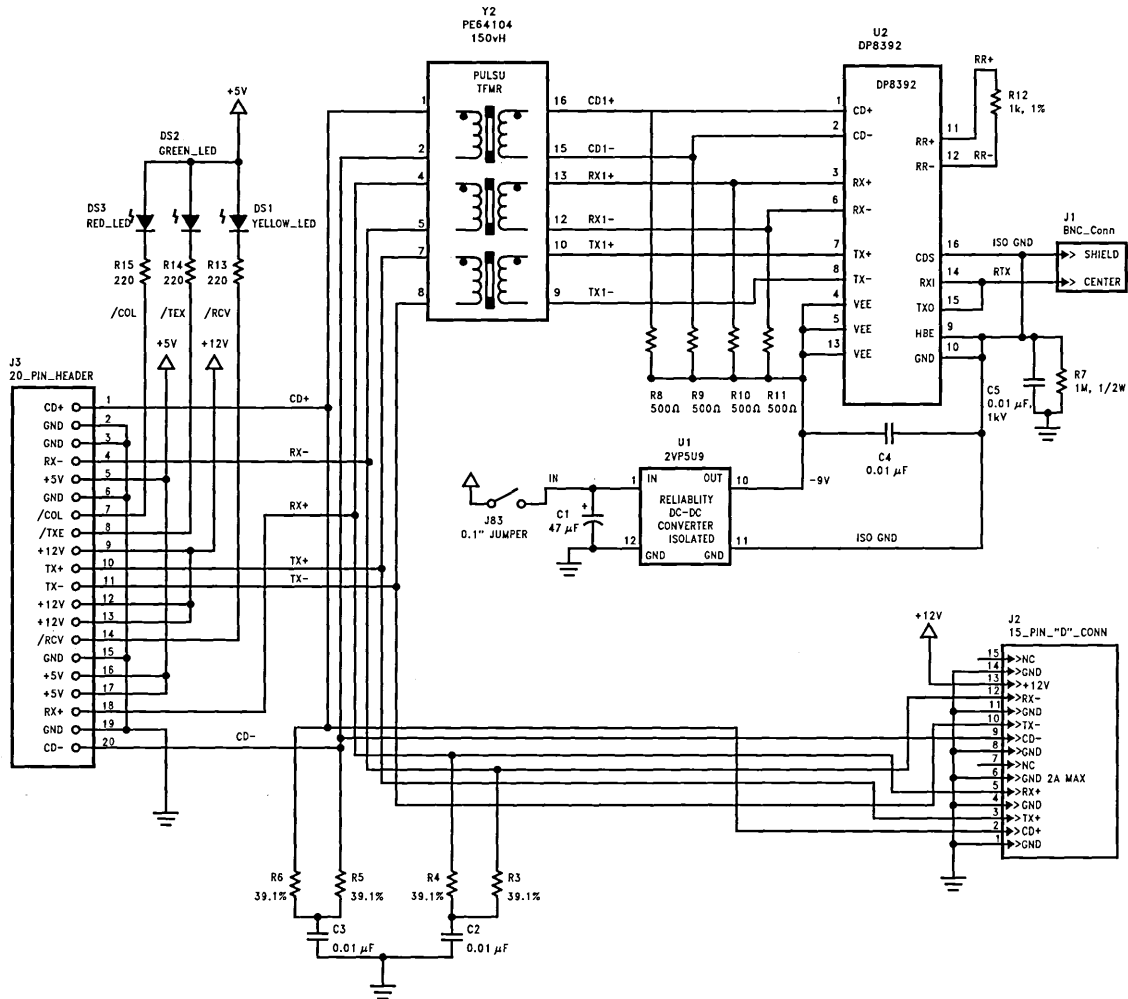
### Equation:

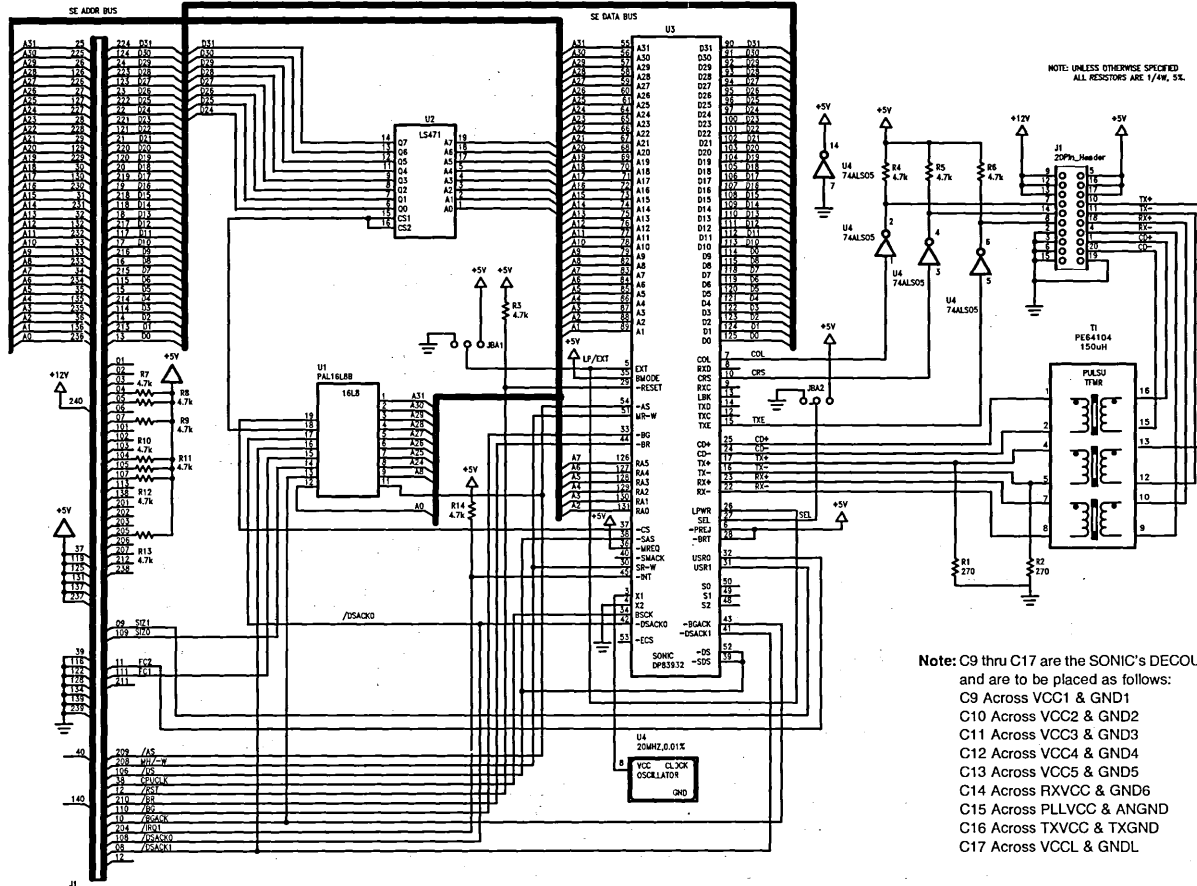
```

A0 = 0
SIZ0 = 0
FC1 = 0
PDSACK0 = 0
PDSACK1 = 1
ENABLE A0 = -BGACK
ENABLE SIZ0 = -BGACK
ENABLE FC1 = -BGACK
ENABLE PDSACK0 = A31*A30*A29*A28*A27*-A26*-A25*A24*A8*-AS
ENABLE PDSACK1 = A31*A30*A29*A28*A27*-A26*-A25*A24*A8*-AS
SONICCS = A31*A30*A29*A28*A27*-A26*-A25*A24*-A8*-AS
PROMSEL = A31*A30*A29*A28*A27*-A26*-A25*A24*A8*-AS

```



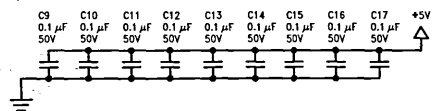
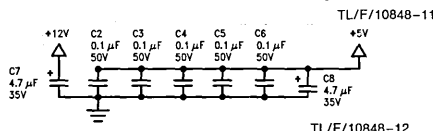




NOTE: UNLESS OTHERWISE SPECIFIED  
ALL RESISTORS ARE 1/4W, 5%.

Note: C9 thru C17 are the SONIC'S DECOUPLING CAPS  
and are to be placed as follows:  
C9 Across VCC1 & GND1  
C10 Across VCC2 & GND2  
C11 Across VCC3 & GND3  
C12 Across VCC4 & GND4  
C13 Across VCC5 & GND5  
C14 Across RXVCC & GND6  
C15 Across PLLVCC & ANGND  
C16 Across TXVCC & TXGND  
C17 Across VCCL & GNDL

Note: J1 Pins A1-A40 = 01-40  
Pins B1-B40 = 101-140  
Pins C1-C40 = 201-240



# DP839EB-MCS SONIC™ MicroChannel® Ethernet Adapter

National Semiconductor  
Application Note 732  
Wesley Lee  
Richard Bowers



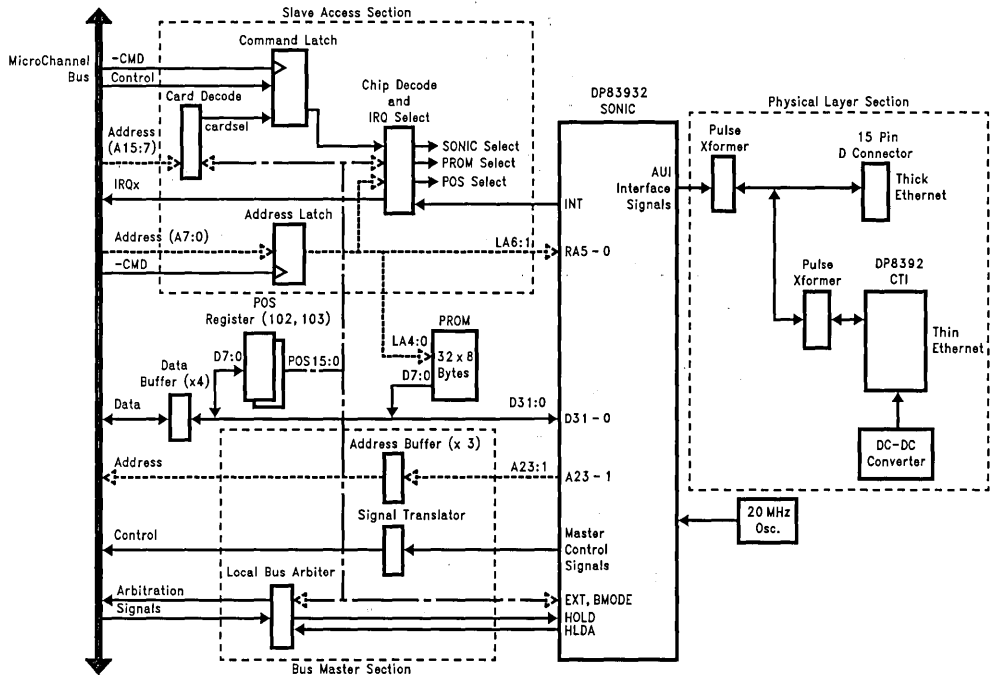
## GENERAL DESCRIPTION

The MicroChannel Evaluation board is a high-performance Ethernet Adapter card designed to provide significant throughput increases over other Ethernet adapter cards currently available. This board employs a bus master architecture for directly storing/retrieving data into system memory; thus, eliminating the need for intermediate packet copying. Using the DP83932's high-speed DMA capabilities, this board achieves a 16 Mbytes/sec transfer rate across the bus, utilizing a 32-bit data and 24-bit address path. The board also features extensive evaluation options to choose between Ethernet/Thin-Ethernet, 16/32-bit data path, and internal/external Ethernet ENDEC. The MicroChannel Ethernet Adapter couples National's DP83932 Systems-Oriented Network Interface Controller (SONIC) with the DP8392 Coaxial Transceiver Interface (CTI) to form a simple two chipset solution for IEEE 802.3 networks.

## FEATURES

- Utilizes the DP83932 and DP8392 chipset
- 16/32-bit data path
- 16 Mbytes/sec DMA throughput
- Extensive SONIC evaluation options
- 14 selectable I/O address ranges
- Compatible with PS/2® models 50, 60, 70, 80
- 4 selectable interrupts
- Bus master/burst capability
- Ethernet/thin-ethernet selectable

## BLOCK DIAGRAM



TL/F/10748-1

## 1.0 BOARD DESCRIPTION

The MicroChannel Ethernet Adapter is a high-performance, 16/32-bit busmaster board designed to demonstrate the advanced features of the DP83932. It consists of three main sections: (1) the physical layer, (2) the slave access section, and (3) the bus master section. The Physical Layer section consists of the internal ENDEC (encoder/decoder) of the SONIC, the coaxial transceiver (DP8392) and isolation transformers and is responsible for driving and receiving the IEEE 802.3 networks signals. The Slave Access section consists of the address decode circuitry necessary to access the SONIC's internal registers, the POS registers, and the PROM. Finally, the Bus Master section composes of the interface logic which permits the SONIC to gain access of the MicroChannel bus.

## 2.0 PHYSICAL LAYER SECTION

The physical layer section drives and encodes data onto the network during transmission, and decodes the data during reception. The Ethernet Adapter uses the on-chip ENDEC (encoder/decoder) from the SONIC and the coaxial transceiver, the DP8392, for these purposes. This physical layer section is illustrated in *Figure 1*.

The Ethernet Adapter provides connections to both thick-wire and thin-wire Ethernet. The thick-wire (AUI) connection is made via the TX $\pm$ , RX $\pm$ , and CD $\pm$  signals from the SONIC, an external pulse transformer and a 15-pin D connector. The external pulse transfer is required to meet the IEEE 802.3 high voltage (16V) specification at AUI interface. The thin-wire connection is made via another external pulse transformer and the on-board coaxial transceiver, the DP8392. This external pulse transformer is necessary to completely isolate the TX $\pm$ , RX $\pm$ , and CD $\pm$  pins when the DP8392 is powered down.

## 2.1 Switching between Thin or Thick-Wire Ethernet

To swap between thin and thick-wire Ethernet, POS 10-bit turns on/off the DC-DC converter. In the thin-wire configuration, the DC-DC converter is turned on, powering up the DP8392 to receive/transmit data onto the thin-wire cable. In the thick-wire configuration, the DC-DC converter is turned off, forcing the DP8392 to shut down. The pulse trans-

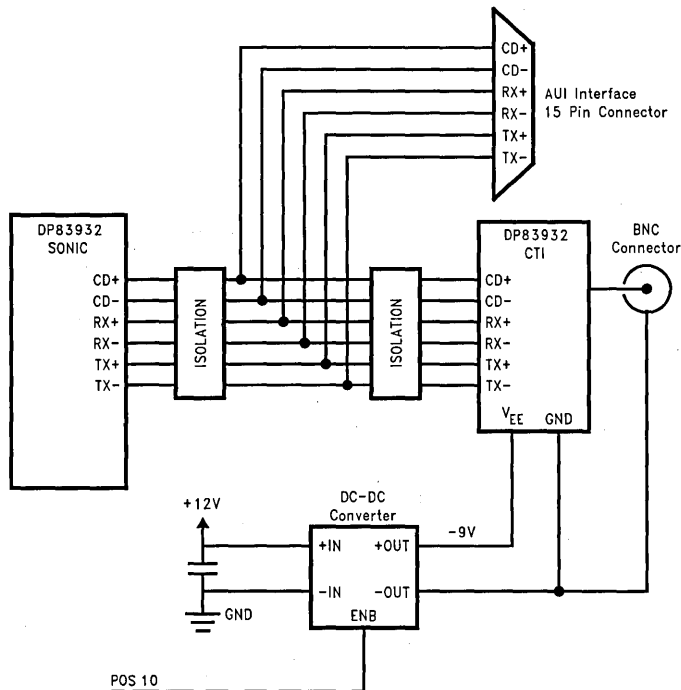


FIGURE 1. Simplified Physical Layer Section

TL/F/10748-2

former B isolates the TX±, RX± and CD± signals of the DP8392 and allows data to pass through the 15-pin D connector unloaded by the DP8392.

**3.0 SLAVE ACCESS SECTION**

During initialization and status updates, the SONIC register may be accessed to provide configuration and status information. The CPU accesses these registers by driving the proper address on the register address lines (RA5-RA0) and chip selecting the SONIC. The SONIC responds with -RDY0 (ready out) when its registers are available for accessing. The CPU may also access the ID PROM and POS registers to respectively read the board's Ethernet Physical Address and configuration information. The system accesses the SONIC, PROM, and POS register via the Address Decode Logic described below. (The slave access section is shown on page 1 of the schematic.)

**3.1 Address Decode Logic**

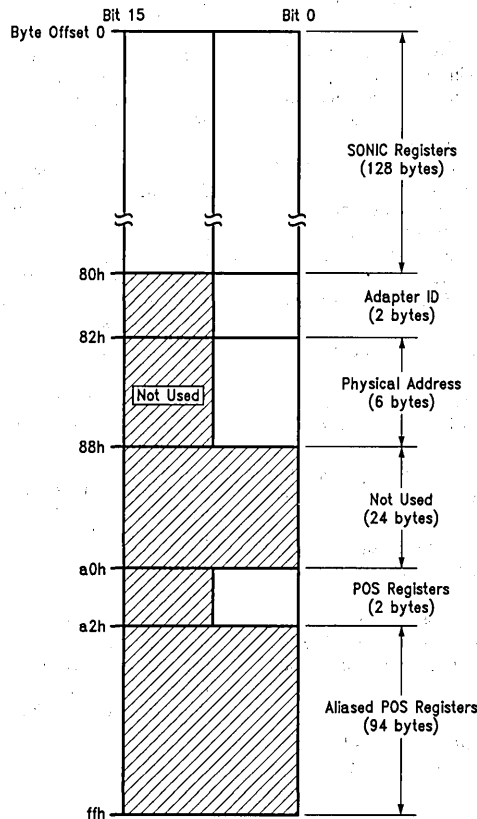
The Address Decode Logic provides the decoding for the SONIC, POS registers, and PROM. This decoding is user programmable with the Address Select bits (ADDR0-3) from POS register 102. The Decode Logic decodes these bits along with the address and control signals from the MicroChannel bus to determine the selected address base (see POS register 102 description in Section 5.0). The de-

coding is a two step process. First, the unlatched address and control signals are decoded to determine if the card is selected; then further decoding is performed to select the proper component on the board. Because the address and control signal on MicroChannel bus are not valid for the duration of the bus cycle, these signals must be latched.

The Address Decode Logic, implemented with two PALs and two latches, operates in the following manner. The CARD DECODE PAL first decodes the unlatched address (A15-A7) and control signal (M/-IO) from the MicroChannel bus along with the POS address select bits (ADDR0-3) to generate the CARDESEL signal. This signal is then latched and further decoding is performed by the CHIP DECODE PAL. This PAL decodes the latched -LCARDESEL signal, the latched lower address LA0-2, LA5-7 and the latched control signals (-LS0, -LS1, and -LCDSETUP) to provide chip selects the SONIC, POS registers, or the PROM. The address and control signals are latched on the leading edge of -CMD with a ACT573 transparent latch.

**3.2 I/O ADDRESS MAP**

The MicroChannel Ethernet Adapter's address base is specified by bits ADDR3-0 in POS register 102. The board occupies 256 bytes as shown in *Figure 2*.



**FIGURE 2. Ethernet Adapter's I/O Address Map**

TL/F/10748-3

## 4.0 BUS MASTER SECTION

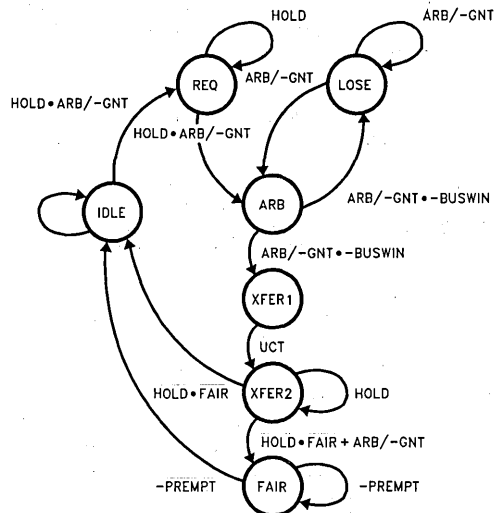
The MicroChannel evaluation board employs a bus master architecture to quickly transfer data from/to system memory to/from the internal FIFOs of the DP83932. The DP83932's FIFOs are sufficiently large (32 bytes) to absorb all reasonable bus latencies which may occur on the MicroChannel bus. When accessing the bus, the evaluation board follows the required MicroChannel protocol using the arbitration level programmed in the POS 103 register. The bus master logic is partitioned into two sections, the Local Bus Arbiter which implements the MicroChannel bus access protocol and the Signal Translator which interfaces the SONIC control signals to the MicroChannel bus. (The bus master section is shown on page 2 of the schematic.)

### 4.1 Local Bus Arbiter

The local bus arbiter allows the Ethernet Adapter to compete for the MicroChannel bus when an arbitration cycle is in progress. The arbitration cycle begins after the Ethernet Adapter asserts -PREMPT and the ARB/-GNT signal has subsequently gone high. The Ethernet adapter participates in the arbitration cycle by gating its arbitration vector onto the bus and simultaneously comparing it with all other vectors appearing on the bus. If the Local Arbiter detects an arbitration vector lower than its own, it removes its vector and waits for the next arbitration cycle; otherwise, if the Ethernet Adapter has won, it may begin transferring data onto the bus.

The Local Arbiter is implemented with two PALs, ARBVEC, and ARBMAC (support logic is also in the Logic PAL). The ARBVEC PAL implements the arbitration vector function which drives and simultaneously reads the vector on lines ARB3-ARB0 when an arbitration cycle is in progress. This PAL indicates whether it has won the bus by asserting -BUSWIN low. The arbitration vector is user programmable via bits SELARBO-3 of POS register 103. The second PAL, ARBMAC, controls the enabling of the arbitration vector. This PAL consists of an asynchronous state machine to monitor bus activity from the MicroChannel bus and the SONIC. This state machine shown in *Figure 3* contains the following states.

- IDLE:** No bus activity by the SONIC; HOLD request is not asserted
- REQ:** The SONIC is requesting usage of the bus but another device may be using the bus; -PREMPT is asserted
- ARB:** An arbitration cycle is occurring on the bus. The arbitration vector is enabled; -PREMPT is still asserted.
- LOSE:** The SONIC has lost the arbitration cycle. It de-gates its vector and waits for the next cycle. -PREMPT is still asserted.
- XFER1:** The SONIC has won; detects -BUSWIN low from ARBVEC PAL. An intermediate state to XFER2. (This state is needed so that only one output changes between states.)  
PREMPT is still asserted.
- XFER2:** The SONIC performs its data transfer. -BURST is asserted and -HLDA is issued to the SONIC. -PREMPT is deasserted.
- FAIR:** The fairness algorithm has been enabled.



Note: UCT = Unconditional Transfer

TL/F/10748-4

**FIGURE 3. Local Bus Arbiter State Machine (ARBMAC)**

The Local Arbiter state machine defaults to IDLE when the SONIC is idle. When HOLD is asserted, the state machine transitions to REQ and then to ARB when ARB/-GNT goes high. During this state, ARBMAC enables ARBVEC to begin driving its vector onto lines ARB3-0 and monitors -BUSWIN when ARB/-GNT has subsequently gone low. If -BUSWIN is high, the Ethernet adapter has lost and the state machine proceeds to LOSE and waits for the next arbitration cycle. Otherwise, the Ethernet adapter has won and the state machine proceeds to XFER2 via XFER1. During XFER2, Hold Acknowledge (HLDA) is issued to the SONIC, allowing it to transfer data onto the bus. (For robust asynchronous state machine design, XFER1 is used to insure that only one output changes between states.) When the SONIC finishes its block transfer, the state machine finally transitions either to FAIR or IDLE, depending upon the FAIR bit in the POS 103 register.

The state machine obeys the fairness algorithm when the FAIR bit in POS register 103 has been set. This algorithm insures that all competing devices will eventually gain access to the bus. If the FAIR bit is set, the state machine will stay in FAIR until all other devices have completed their bus transfers (i.e., -PREMPT is no longer asserted) before returning to IDLE.

### 4.2 Signal Translator

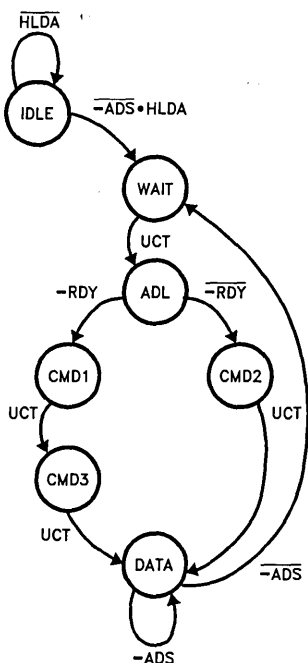
The Signal Translator converts the control signals from the SONIC into the signals required by the MicroChannel bus and matches the timing of these signals to be consistent with the 3 primary modes of the MicroChannel specification, *Default mode*, *Synchronous Extended mode*, and *Asynchronous Extended mode*. The Signal Translator matches the

timing by using a synchronous state machine to convert the SONIC's control signals, MW-R, -ADS, and -DS to -S0, -S1, -ADL, and -CMD of the MicroChannel bus. Note that since the SONIC is capable of operating considerably faster than the MicroChannel bus, it is required that the SONIC be programmed in asynchronous mode and inserts 3 wait-states for each memory cycle. Configuring asynchronous mode and the 3 wait-states are programmed by resetting the STERM bit and setting bits WC1 and WC0 and in the Data Configuration register. The state machine, shown in Figure 4, contains the following states.

**IDLE:** No bus activity by the SONIC.

**WAIT:** SONIC has asserted -ADS; wait for one bus clock to provide address setup time to leading edge of -ADL.

**ADL:** Assert -ADL for one clock cycle.



Note: UCT = Unconditional Transfer

TL/F/10748-5

Note: The state machine is reset when HLDA is deasserted.

**FIGURE 4. Signal Translator State Machine**

**CMD1:** *Synchronous or Asynchronous Extended* mode requested. CDCHRDY has been deasserted by memory. Begin asserting -CMD and SYNWAIT.

**CMD2:** *Default* mode requested. Begin asserting CMD.

**CMD3:** Continuing to assert -CMD and SYNWAIT.

**DATA:** SONIC finishes up memory transfer. -CMD is deasserted when -DS goes low. Transition back to IDLE when -ADS goes low.

The Signal Translator may take one of two paths during a SONIC bus operation. In the first path, the Signal Translator goes through states, WAIT, ADL, CMD2, and finally DATA. This path is taken if the memory does not require any wait-states and allows the SONIC to operate as fast as permitted on the MicroChannel bus (*Default* mode, minimum cycle time = 200 ns). To satisfy all the timing requirements of MicroChannel, the SONIC must use 5 bus clocks at 20 MHz for the memory cycle. This path is illustrated in Figure 5. The second path goes through states: WAIT, ADL, CMD1, CMD3, and DATA. This path gives at least an additional 100 ns to the memory cycle for compatibility with the *Synchronous Extended* mode (300 ns) or the *Asynchronous Extended* mode ( $\geq 300$  ns).

The memory may deassert CDCHRDY in two ways. In the first manner, the memory requests a *Synchronous Extended* mode by pulsing CDCHRDY within 60 ns after the address goes valid then driving it active after -CMD has subsequently gone low. The timing is illustrated in Figure 6. (Note that the Ethernet adapter accesses the ready signal via the return signal, CHRDRYRTN.) The additional 100 ns is added by deasserting the RDYi input of the SONIC during CMD1 and CMD3. The RDYi input is deasserted by NANDing (in the IRQSEL PAL) the SYNWAIT output generated by the Signal Translator with CHRDRYRTN. Note that in asynchronous mode, the SONIC terminates the memory cycle 1 bus after clock after -RDYi is asserted.) In the second manner, the memory requests the *Asynchronous* mode by deasserting CDCHRDY as before, but does not assert CDCHRDY until it is ready to be accessed.

In addition to the signals generated by the Signal Translator (-S0, S1, -ADL, -CMD), there are 8 other signals which must be generated each time the SONIC gains access to the bus. These signals are MADE24, M/-IO, -SHBE, TR32, and BE0-3. The first 4 signals are enabled whenever the SONIC is bus master, and the latter 4 are enabled when the SONIC is configured for 32-bit data mode. Since these signals remain constant for the duration of the SONIC's transfer cycle, they are driven to their proper levels using a ACT244.

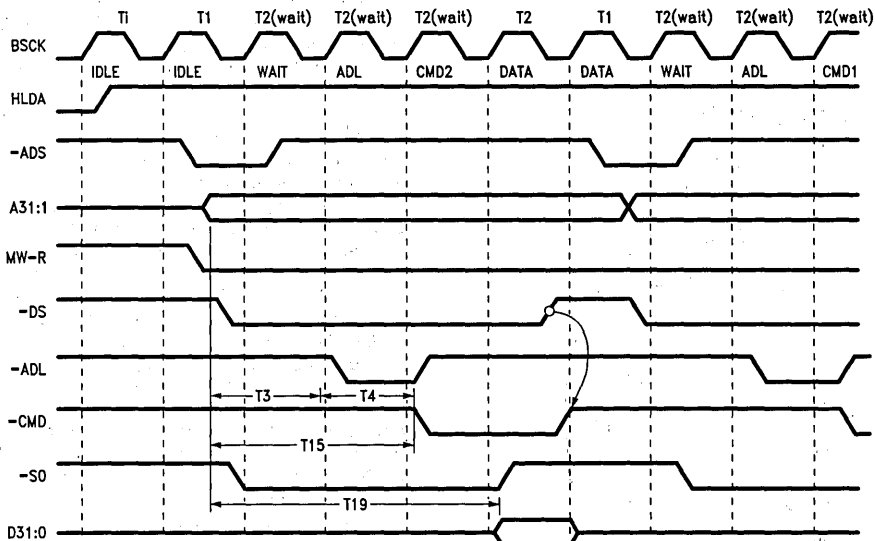
**4.2.1 SIGNAL TRANSLATOR TIMING**

Figures 5, 6, and 7 show the timing generated by the Signal translator for Default Mode (minimum cycle time = 200 ns), Synchronous Extended Mode (minimum cycle time = 300 ns) and Asynchronous mode (cycle time ≥ 300 ns). States Ti, T1 and T2 indicate the DMA states of the SONIC while states IDLE, WAIT, ADL, CMD1, CMD2, CMD3, and DATA indicate the corresponding states of the Signal Translator. Note that the SONIC must be configured in asynchronous mode and be inserting 3 wait-states. Also note that CHRDRYRTN, shown in Figures 6 and 7, is the ready return signal provided by the MicroChannel bus.

The timing parameters, shown as "T #," indicate the critical timing constraints which the Signal Translator and the SONIC must meet in order to be compatible with the

MicroChannel bus. These parameters and the corresponding Ethernet adapter parameters are tabulated below.

Parameter	MicroChannel Spec.	Ethernet Adapter
T3	45 ns (min)	75 ns (min)
T4	40 ns (min)	50 ns (min)
T15	85 ns (min)	125 ns (min)
T19	125 ns (max)	200 ns (max)
T26	60 ns (max)	100 ns (max)
T28D	160 ns (max)	175 ns (max)
T29S	60 ns (max)	60 ns (max)



**FIGURE 5. Default Mode, Memory Read**

TL/F/10748-6



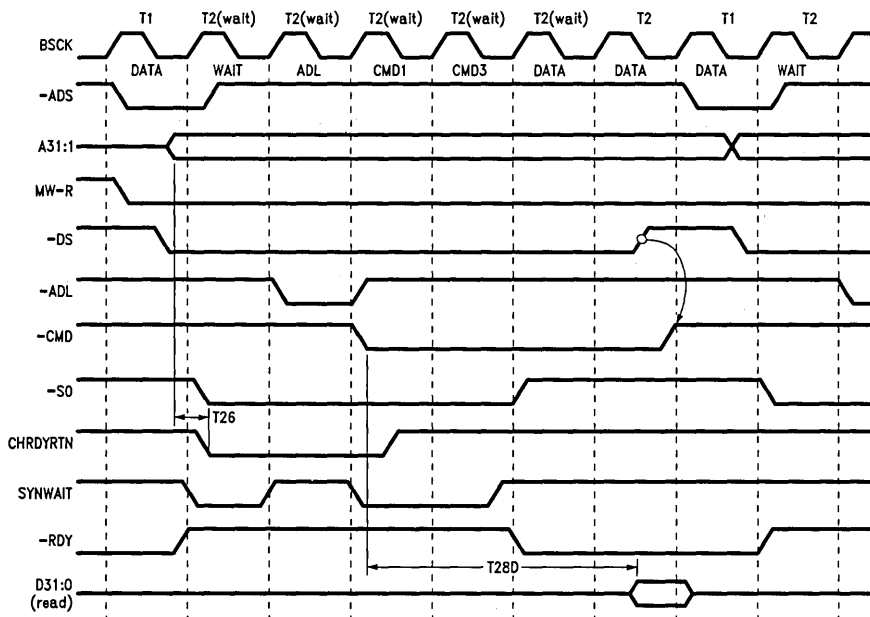


FIGURE 6. Synchronous Extended Mode, Memory Read

TL/F/10748-7

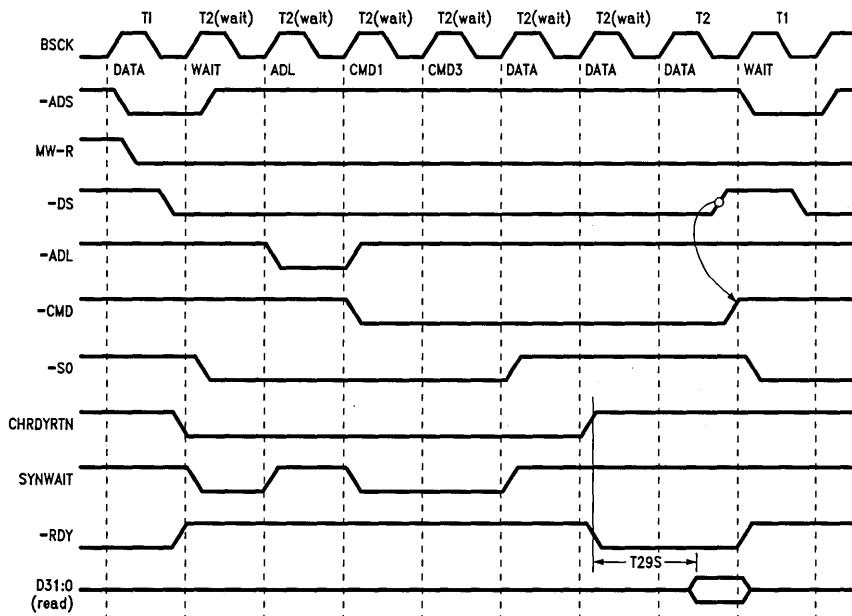


FIGURE 7. Asynchronous Extended Mode, Memory Read

TL/F/10748-8

### 5.0 POS REGISTERS

The POS registers provide the required identification (ID) bytes (2) and configuration information needed by the Ethernet Adapter. The MicroChannel bus during power-on reads the ID bytes, located at addresses 100 and 101, and compares these bytes with values saved in battery back-up RAM. If the comparison is true, it will proceed to write the configuration information stored in RAM into the remaining

POS registers residing at addresses 102 to 103. The configuration information, guarantees that no I/O, memory, or interrupts conflict with other boards using the MicroChannel bus. For this implementation, only two POS registers are needed to provide the required configuration parameters. The following description gives the bit definitions for POS registers 102 and 103.

#### POS 102 BIT DEFINITIONS

DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
ADDR3	ADDR2	ADDR1	ADDR0	res	INT1	INT0	CARDEN

Bit	Name	Function																																		
7-4	ADDR3-0	<p><b>Address Select:</b> These bits select which base address the Ethernet Adapter Card will reside. The address selections are shown below:</p> <table border="1"> <thead> <tr> <th>ADDR3-0</th> <th>I/O Address</th> </tr> </thead> <tbody> <tr><td>1111</td><td>Not Used</td></tr> <tr><td>1110</td><td>Not Used</td></tr> <tr><td>1101</td><td>1e00-1effh</td></tr> <tr><td>1100</td><td>1c00-1cffh</td></tr> <tr><td>1011</td><td>1a00-1affh</td></tr> <tr><td>1010</td><td>1800-18ffh</td></tr> <tr><td>1001</td><td>1600-16ffh</td></tr> <tr><td>1000</td><td>1400-14ffh</td></tr> <tr><td>0111</td><td>1200-12ffh</td></tr> <tr><td>0110</td><td>1000-10ffh</td></tr> <tr><td>0101</td><td>0e00-0effh</td></tr> <tr><td>0100</td><td>0c00-0cffh</td></tr> <tr><td>0011</td><td>0a00-0affh</td></tr> <tr><td>0010</td><td>0800-08ffh</td></tr> <tr><td>0001</td><td>0600-06ffh</td></tr> <tr><td>0000</td><td>0400-04ffh</td></tr> </tbody> </table>	ADDR3-0	I/O Address	1111	Not Used	1110	Not Used	1101	1e00-1effh	1100	1c00-1cffh	1011	1a00-1affh	1010	1800-18ffh	1001	1600-16ffh	1000	1400-14ffh	0111	1200-12ffh	0110	1000-10ffh	0101	0e00-0effh	0100	0c00-0cffh	0011	0a00-0affh	0010	0800-08ffh	0001	0600-06ffh	0000	0400-04ffh
ADDR3-0	I/O Address																																			
1111	Not Used																																			
1110	Not Used																																			
1101	1e00-1effh																																			
1100	1c00-1cffh																																			
1011	1a00-1affh																																			
1010	1800-18ffh																																			
1001	1600-16ffh																																			
1000	1400-14ffh																																			
0111	1200-12ffh																																			
0110	1000-10ffh																																			
0101	0e00-0effh																																			
0100	0c00-0cffh																																			
0011	0a00-0affh																																			
0010	0800-08ffh																																			
0001	0600-06ffh																																			
0000	0400-04ffh																																			
3	res	<b>Reserved</b>																																		
2-1	INT1,0	<p><b>Interrupt Select:</b> These bits select the Interrupt Request lines: The selections are shown below:</p> <table border="1"> <thead> <tr> <th>INT1,0</th> <th>IRQ line</th> </tr> </thead> <tbody> <tr><td>11</td><td>IRQ9</td></tr> <tr><td>10</td><td>IRQ7</td></tr> <tr><td>01</td><td>IRQ6</td></tr> <tr><td>00</td><td>IRQ3</td></tr> </tbody> </table>	INT1,0	IRQ line	11	IRQ9	10	IRQ7	01	IRQ6	00	IRQ3																								
INT1,0	IRQ line																																			
11	IRQ9																																			
10	IRQ7																																			
01	IRQ6																																			
00	IRQ3																																			
0	CARDEN	<p><b>Card Enable:</b> When this bit is set to a "0", the card is disabled and responds only to the setup read and write commands. When set to a "1" the card is enabled.</p>																																		

## POS 103 BIT DEFINITIONS

DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
ARB3	ARB2	ARB1	ARB0	FAIR	E/T	EXT	res

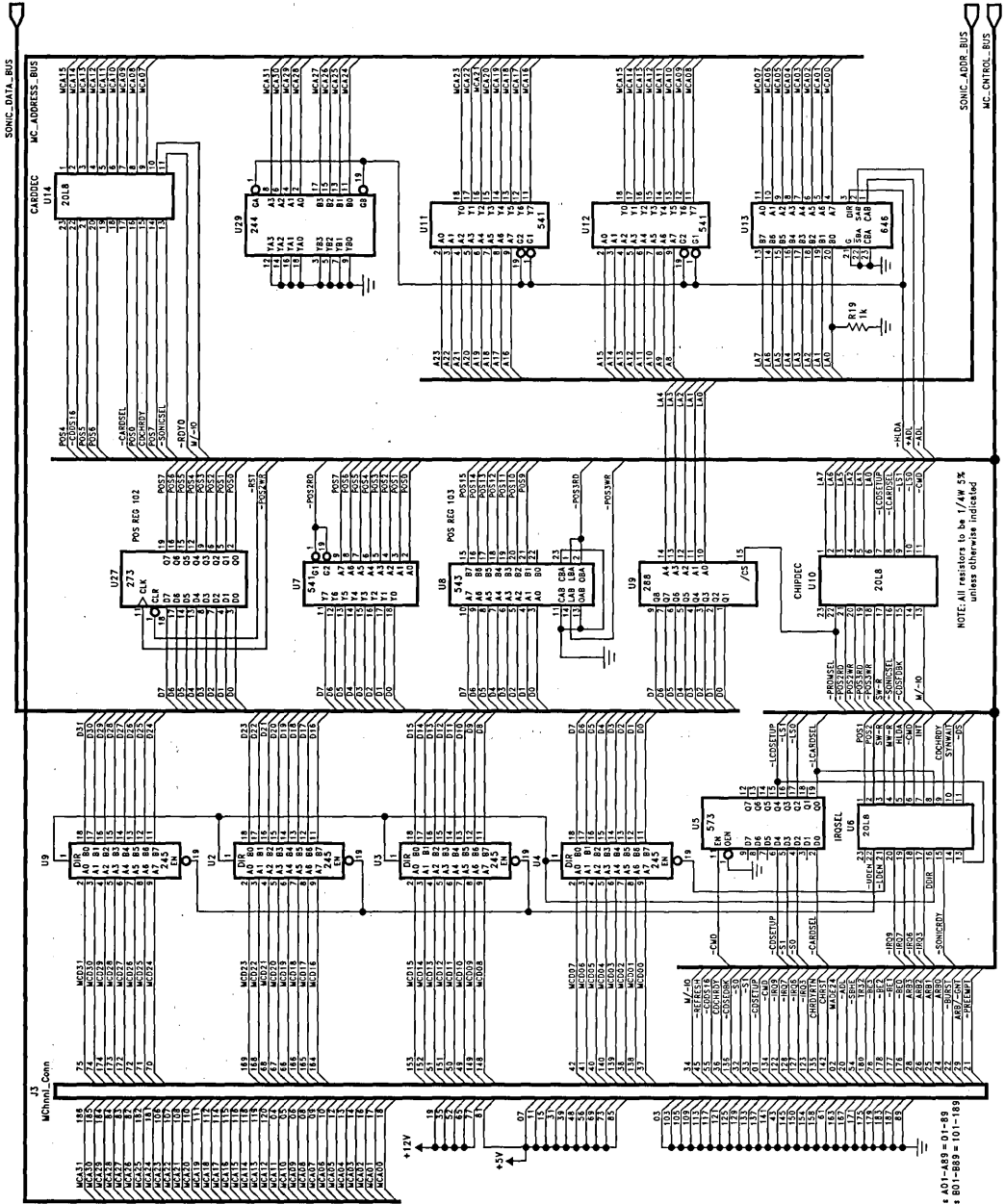
Bit	Name	Function																																																			
7-4	SELARB3-0	<p><b>Arbitration Vector:</b> These bits select the arbitration vector when SONIC contends for the bus. The selections are shown below:</p> <table border="0"> <tr> <td><b>SELARB3-0</b></td> <td><b>Arbitration Vector</b></td> <td></td> </tr> <tr> <td>1111</td> <td>Not Used</td> <td></td> </tr> <tr> <td>1110</td> <td>ARB14</td> <td>Lowest Priority</td> </tr> <tr> <td>1101</td> <td>ARB13</td> <td></td> </tr> <tr> <td>1100</td> <td>ARB12</td> <td></td> </tr> <tr> <td>1011</td> <td>ARB11</td> <td></td> </tr> <tr> <td>1010</td> <td>ARB10</td> <td></td> </tr> <tr> <td>1001</td> <td>ARB9</td> <td></td> </tr> <tr> <td>1000</td> <td>ARB8</td> <td></td> </tr> <tr> <td>0111</td> <td>ARB7</td> <td></td> </tr> <tr> <td>0110</td> <td>ARB6</td> <td></td> </tr> <tr> <td>0101</td> <td>ARB5</td> <td></td> </tr> <tr> <td>0100</td> <td>ARB4</td> <td></td> </tr> <tr> <td>0011</td> <td>ARB3</td> <td></td> </tr> <tr> <td>0010</td> <td>ARB2</td> <td></td> </tr> <tr> <td>0001</td> <td>ARB1</td> <td></td> </tr> <tr> <td>0000</td> <td>ARB0</td> <td>Highest Priority</td> </tr> </table>	<b>SELARB3-0</b>	<b>Arbitration Vector</b>		1111	Not Used		1110	ARB14	Lowest Priority	1101	ARB13		1100	ARB12		1011	ARB11		1010	ARB10		1001	ARB9		1000	ARB8		0111	ARB7		0110	ARB6		0101	ARB5		0100	ARB4		0011	ARB3		0010	ARB2		0001	ARB1		0000	ARB0	Highest Priority
<b>SELARB3-0</b>	<b>Arbitration Vector</b>																																																				
1111	Not Used																																																				
1110	ARB14	Lowest Priority																																																			
1101	ARB13																																																				
1100	ARB12																																																				
1011	ARB11																																																				
1010	ARB10																																																				
1001	ARB9																																																				
1000	ARB8																																																				
0111	ARB7																																																				
0110	ARB6																																																				
0101	ARB5																																																				
0100	ARB4																																																				
0011	ARB3																																																				
0010	ARB2																																																				
0001	ARB1																																																				
0000	ARB0	Highest Priority																																																			
3	FAIREN	<p><b>Fairness Enable:</b> When this bit is set to a "1", the MicroChannel fairness algorithm is used. When set to a "0", fairness is disabled.</p>																																																			
2	E/T	<p><b>Ethernet/Thin-Ethernet Select:</b> This bit selects between the Ethernet and Thin-Ethernet options. This pin is directly connected to the DC-DC Converter.</p> <p>0: Thin-Ethernet selected (Thin cable) 1: Ethernet selected (Thick cable)</p>																																																			
1	EXT	<p><b>External ENDEC Select:</b> This bit selects between the internal and external ENDEC options. This bit is directly connected to the EXT pin of the SONIC.</p> <p>0: The SONIC's internal ENDEC is enabled. 1: The internal ENDEC of the SONIC is disabled. An external ENDEC is to be used.</p>																																																			
0	res	<b>Reserved</b>																																																			

## PARTS LIST

Qty.	Part	Number
1	0.01 $\mu$ F/50V	C2
1	0.01 $\mu$ F/500V	C1
2	0.01 $\mu$ F/50V	C4, C5
1	47 $\mu$ F	C3
1	0.001 $\mu$ F	C6
36	0.1 $\mu$ F	C10-C46
2	4.7 $\mu$ F	C47, C48
1	1N4150	D1
2	1K, 1%	R1, R19
1	1M, $\frac{1}{2}$ w	R2
4	1.5K	R3, R4, R5, R6
4	39.2, 1%	R7, R8, R9, R10
2	270	R11, R12
5	4.7K	R14, R15, R16, R17, R18
2	PE64104	T1, T2
1	DP83932 SONIC	U19
1*	DP83910 SNI	U23
1	DP8392 CTI	U25
2	74ACT244	U15, U29
4	74ACT245	U1, U2, U3, U4
1	74ACT273	U27
1	74S288	U9
3	74ACT541	U7, U11, U12
1	74ACT573	U5
1	74ACT646	U13
2	PAL16R4D	U17, U28
5	PAL20L8D	U6, U10, U14, U16, U18
1	20 MHz, 45% - 55%	U20
1*	xx MHz, 45% - 55%	U21
1	PM9005	U26
1	Spark Gap	C49
1	15 Pin D Connector	J1
1	BNC Connector	J2
1	MChannel Conn.	J3
1	3 Pin Jumper	JU1
1	Dip Switch	SW1
17	Test Points	TP16-TP32

\*Optional

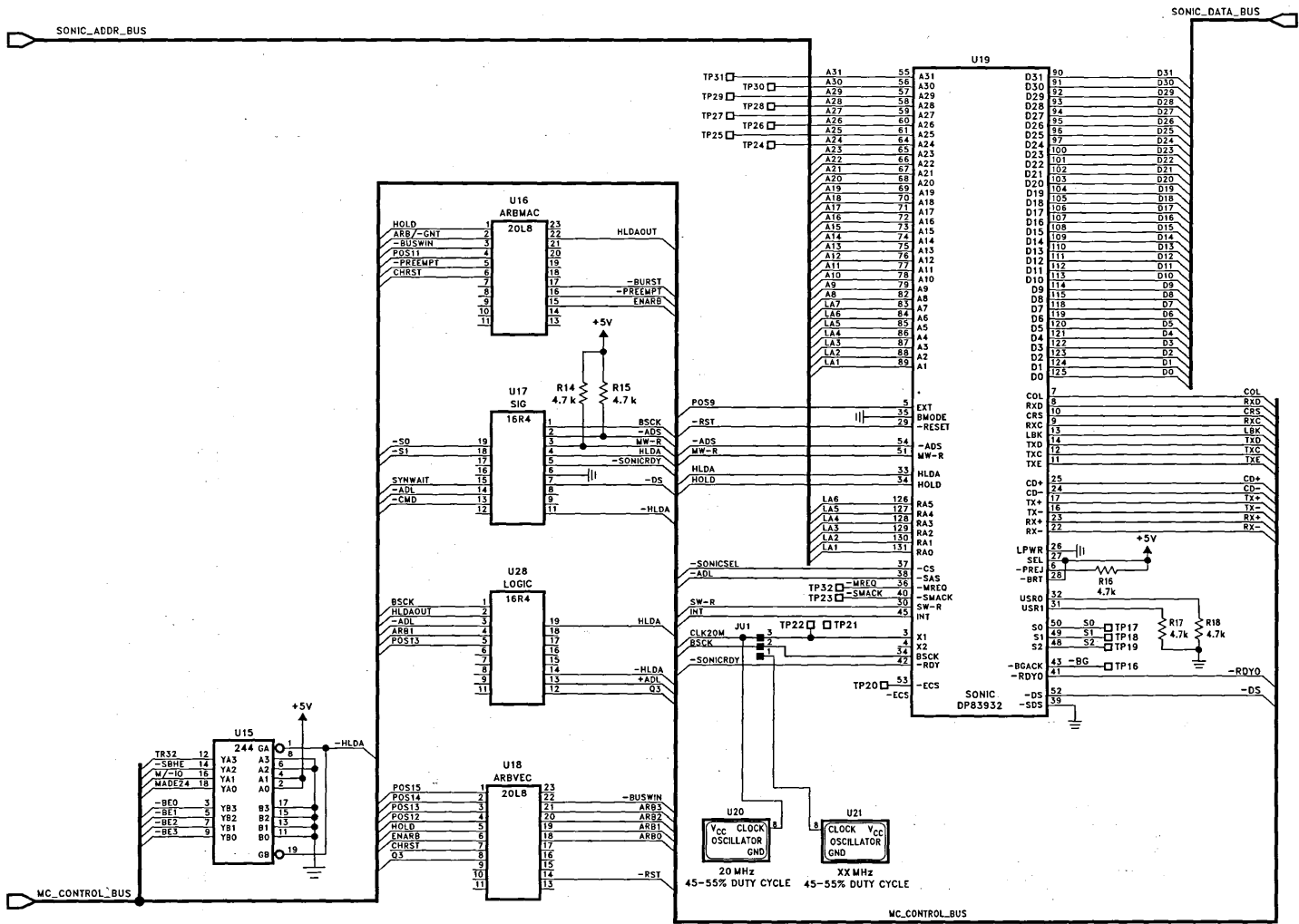
All resistors are 5% unless otherwise specified

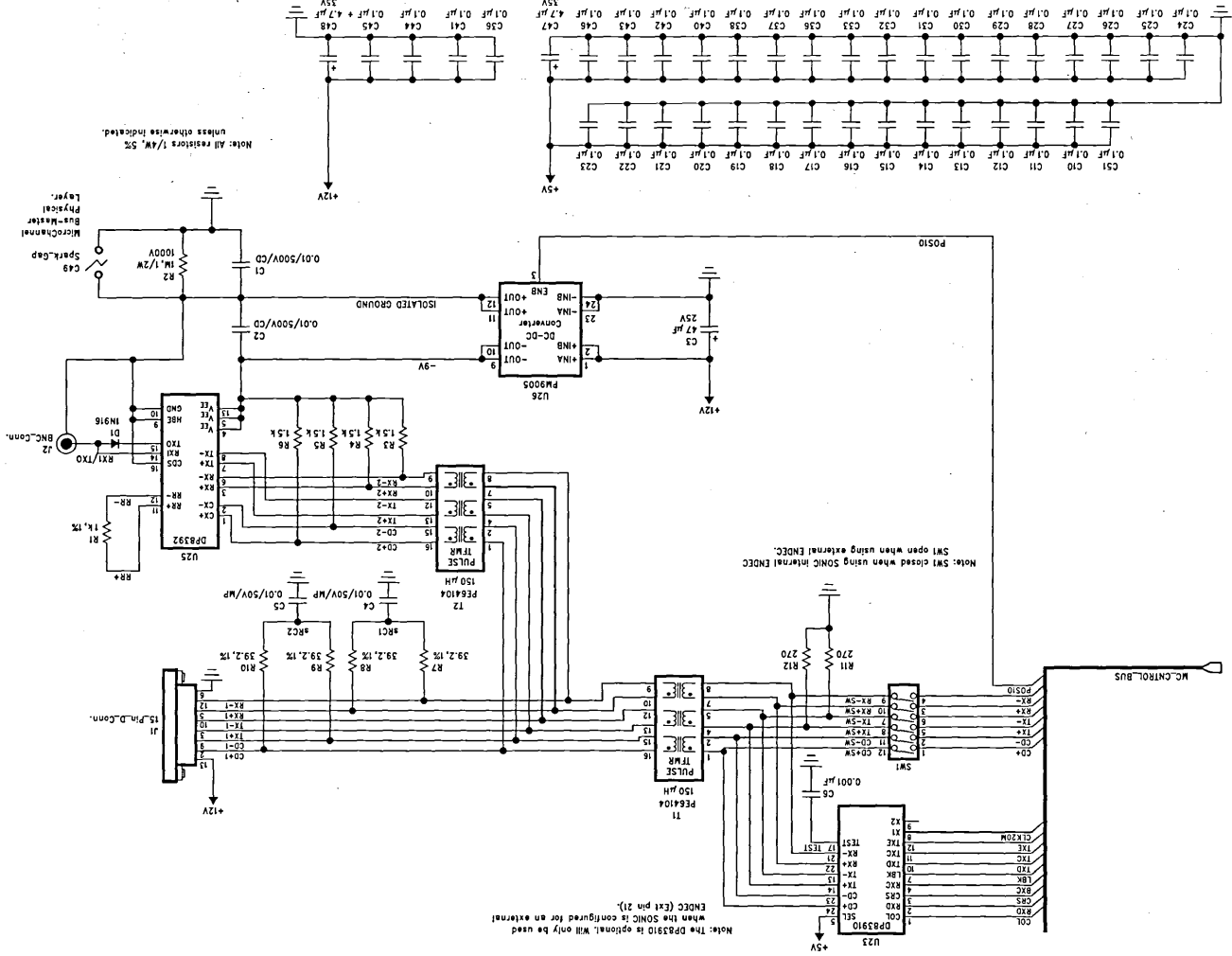


NOTE: All resistors to be 1/4W 5%  
unless otherwise indicating.



1-848





Note: All resistors 1/4W, 5% unless otherwise indicated.

MicroChannel Layer: Bus-Master Sprk.Cap

Note: SW1 closed when using SONIC internal ENDEC. SW1 open when using external ENDEC.

Note: The DB392 is optional. Will only be used when the SONIC is configured for an external ENDEC (exit pin 21).

Note: 0.1µF capacitors to be connected between IC power pins and ground.

## PAL EQUATIONS

The DP83932 SONIC MicroChannel adapter contains 7 PALs to implement the bus interface. This section provides a listing of the PALs used. All listings use the ABEL™ design format.

```

module carddec;          flag '-r2','-t2';

title
'PAL20L8                National Semiconductor
MicroChannel Card Decoder      1/1/90
file name: CARDDEC.ABL                               w1'

CARDDEC device 'P20L8';
" PAL DESCRIPTION

" This pal determines whether the MicroChannel Ethernet adapter
" is selected. The address base is determined by bits 7 - 4
" in POS register 102. The output signals are following:
"   !cardsel = active when base address matches
"   !cdds16 = active when the SONIC register are being accessed
"   !cdchrđy = ready signal

" Equations written in ABEL™ design format.
" declarations

"declarations

TRUE,FALSE = 1,0;
H,L = 1,0;
x,z,c = .X.,.Z.,.C.;

GND,VCC
pin 12,24;

"outputs
halfsel1,halfsel2,cardsel,cdds16,cdchrđy
pin 19,18,17,22,15;

"inputs
a15,a14,a13,a12,a11,a10,a9,a8,a7,mio,sonicssel,
pos4,pos5,pos6,pos7,pos0,rđyo
pin 1,2,3,4,5,6,7,8,9,10,13,23,21,20,14,16,11;

"equates
addr = [a15,a14,a13,a12,a11,a10,a9,a8];
possel = [pos7,pos6,pos5,pos4];

equations

!halfsel1 = (possel == ^h0) & (addr == ^h4) #
            (possel == ^h1) & (addr == ^h6) #
            (possel == ^h2) & (addr == ^h8) #
            (possel == ^h3) & (addr == ^ha) #
            (possel == ^h4) & (addr == ^hc) #
            (possel == ^h5) & (addr == ^he) #
            (possel == ^h6) & (addr == ^h10);

!halfsel2 = (possel == ^h7) & (addr == ^h12) #
            (possel == ^h8) & (addr == ^h14) #

```



## PAL EQUATIONS (Continued)

```
(possel == ^h9) & (addr == ^h16) #
(possel == ^ha) & (addr == ^h18) #
(possel == ^hb) & (addr == ^h1a) #
(possel == ^hc) & (addr == ^h1c) #
(possel == ^hd) & (addr == ^h1e);
```

```
!cardsel = !halfsel1 & !mio & pos0 #
!halfsel2 & !mio & pos0;
```

```
!cdds16 = !cardsel & !a7;
```

```
!cdchrdy = !halfsel1 & !mio & pos0 & !a7 & rdyo #
!halfsel2 & !mio & pos0 & !a7 & rdyo #
!sonicssel & rdyo;
```

## test\_vectors

```
([addr,possel,mio,pos0] -> [cardsel,cdds16])
```

```
[^h4 , ^h0 , 1 , ,1] -> [ 0 , x ]; "cardsel
[^h6 , ^h1 , 1 , ,1] -> [ 0 , x ];
[^h8 , ^h2 , 1 , ,1] -> [ 0 , x ];
[^ha , ^h3 , 1 , ,1] -> [ 0 , x ];
[^hc , ^h4 , 1 , ,1] -> [ 0 , x ];
[^he , ^h5 , 1 , ,1] -> [ 0 , x ];
[^h10 , ^h6 , 1 , ,1] -> [ 0 , x ];
```

```
[^h12 , ^h7 , 1 , ,1] -> [ 0 , x ]; "cardsel
[^h14 , ^h8 , 1 , ,1] -> [ 0 , x ];
[^h16 , ^h9 , 1 , ,1] -> [ 0 , x ];
[^h18 , ^ha , 1 , ,1] -> [ 0 , x ];
[^h1a , ^hb , 1 , ,1] -> [ 0 , x ];
[^h1c , ^hc , 1 , ,1] -> [ 0 , x ];
[^h1e , ^hd , 1 , ,1] -> [ 0 , x ];
```

```
[^h1e , ^hd , 1 , ,0] -> [ 1 , x ];
[^h1c , ^hd , 1 , ,x] -> [ 1 , x ];
```

## test\_vectors

```
([addr,possel,mio,a7,pos0] -> [cardsel,cdds16])
```

```
[^h1e , ^hd , 1 , ,1,1] -> [ 0 , 1 ]; "cdds16
[^h1e , ^hd , 1 , ,0,1] -> [ 0 , 0 ]; "cdds16
[^h1e , ^hd , 1 , ,0,0] -> [ 1 , 1 ]; "cdds16
```

## test\_vectors

```
([addr,possel,mio,pos0,rdyo,a7] -> [cardsel,cdchrdy])
```

```
[^h4 , ^h0 , 0 , 1 , , 1,0 ] -> [ 0 , 0 ]; "cdchrdy
[^h4 , ^h0 , 0 , 1 , , 0,0 ] -> [ 0 , 1 ]; "cdchrdy
[^h6 , ^h0 , 0 , 1 , , 1,0 ] -> [ 1 , 1 ]; "cdchrdy
[^h4 , ^h0 , 1 , 1 , , 1,0 ] -> [ 1 , 1 ]; "cdchrdy
[^h4 , ^h0 , 0 , 1 , , 1,0 ] -> [ 1 , 1 ]; "cdchrdy
[^h4 , ^h0 , 0 , 0 , , 1,0 ] -> [ 1 , 1 ]; "cdchrdy
[^h4 , ^h0 , 0 , 1 , , 1,1 ] -> [ 0 , 1 ]; "cdchrdy
```

```
end carddec;
```

## PAL EQUATIONS (Continued)

```

module chipdec;          flag '-r2','-t2';

title
PAL20L8                  National Semiconductor
MicroChannel Slave Chip Decoder      1/1/90
file name: CHIPDEC.ABL                                w1'

CHIPDEC device 'P20L8';

" PAL DESCRIPTION

" This pal selects which component on the board is accessed.  See
" DP83932 SONIC MicroChannel application note for I/O mapping.
"declarations

    TRUE,FALSE = 1,0;
    H,L = 1,0;
    x,z,c = .X.,.Z.,.C.;

    GND,VCC
        pin 12,24;

"outputs
    sonicssel,promsel,pos2rd,pos2wr,pos3rd,pos3wr,swr,cdsfdbk
    pin 16 , 22 , 21 , 20 , 19 , 18 ,17 , 15;

"inputs
    la7,la6,la5,la2,la1,la0,lchsetup,lcardsel,ls1,ls0,mio,cmd
    pin 1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 ,10 ,13 , 11;

"equates
    addr = [la7,la6,la5,la2,la1,la0];

equations

!sonicssel = !lcardsel & !la7 & !cmd;

!promsel = !lcardsel & la7 & !la6 & !la5 & ls0 & !ls1 & !cmd #
          !lchsetup & !la2 & !la1 & ls0 & !ls1 & !cmd;

!pos2rd = !lcardsel & la7 & !la6 & la5 & !la0 & ls0 & !ls1 & !cmd#
          !lchsetup & !la2 & la1 & !la0 & ls0 & !ls1 & !cmd;

!pos2wr = !lcardsel & la7 & !la6 & la5 & !la0 & !ls0 & ls1 & !cmd#
          !lchsetup & !la2 & la1 & !la0 & !ls0 & ls1 & !cmd;

!pos3rd = !lcardsel & la7 & !la6 & la5 & la0 & ls0 & !ls1 & !cmd #
          !lchsetup & !la2 & la1 & la0 & ls0 & !ls1 & !cmd;

!pos3wr = !lcardsel & la7 & !la6 & la5 & la0 & !ls0 & ls1 & !cmd #
          !lchsetup & !la2 & la1 & la0 & !ls0 & ls1 & !cmd;

!swr = ls0 & !ls1;

!cdsfdbk = !lcardsel & lchsetup & !mio;

test_vectors
([addr ,lchsetup,lcardsel,ls0,ls1,cmd] -> [sonicssel,promsel,swr])

```

## PAL EQUATIONS (Continued)

```
[^b0111111, x , 0 , , x , x , 0 ] -> [ 0 , x , x ]; "sonicsel
[^b1111111, x , 0 , , x , x , x ] -> [ 1 , x , x ]; "mreq
[^b1111111, x , 0 , , 1 , 0 , x ] -> [ x , x , 0 ]; "swr
[^b1111111, x , 0 , , 0 , 1 , x ] -> [ x , x , 1 ]; "swr

[^b100111, x , 0 , , 1 , 0 , 0 ] -> [ 1 , 0 , x ]; "promsel
[^b101111, x , 0 , , 1 , 0 , 0 ] -> [ 1 , 1 , x ]; "promsel
[^b111001, 0 , x , , 1 , 0 , 0 ] -> [ 1 , 0 , x ];
[^b111000, 0 , x , , 1 , 0 , 0 ] -> [ 1 , 0 , x ];
```

## test\_vectors

```
([addr , lchsetup, lcardsel, ls0, ls1, cmd] -> [pos2rd, pos2wr, pos3rd, pos3wr])
```

```
[^b101000, x , 0 , , 1 , 0 , 0 ] -> [ 0 , 1 , 1 , 1 ]; "pos2rd
[^b111010, 0 , x , , 1 , 0 , 0 ] -> [ 0 , 1 , 1 , 1 ];

[^b101000, x , 0 , , 0 , 1 , 0 ] -> [ 1 , 0 , 1 , 1 ]; "pos2wr
[^b111010, 0 , x , , 0 , 1 , 0 ] -> [ 1 , 0 , 1 , 1 ];

[^b101001, x , 0 , , 1 , 0 , 0 ] -> [ 1 , 1 , 0 , 1 ]; "pos3rd
[^b111011, 0 , x , , 1 , 0 , 0 ] -> [ 1 , 1 , 0 , 1 ];

[^b101001, x , 0 , , 0 , 1 , 0 ] -> [ 1 , 1 , 1 , 0 ]; "pos3wr
[^b111011, 0 , x , , 0 , 1 , 0 ] -> [ 1 , 1 , 1 , 0 ];
```

## test\_vectors

```
([lcardsel, lchsetup, mio] -> [cdsfdbk])
```

```
[ 1 , x , x ] -> [ 1 ];
[ x , 0 , x ] -> [ 1 ];
[ 0 , 1 , 0 ] -> [ 0 ];
```

```
end chipdec;
```

TL/F/10748-15

## PAL EQUATIONS (Continued)

```

module irqsel;          flag '-r2','-t2';

title
'PAL20L8                National Semiconductor
MicroChannel IRQ Selector and Buffer Enable          1/1/90
file name: IRQSEL.ABL                                wl'

IRQSEL device 'P20L8';

" This pal selects which interrupt line to use (IRQ9, IRQ7, IRQ6, or
" IRQ3) when the SONIC asserts its interrupt. Each IRQ line is an open
" collector type output (only asserted low). This pal also enables the
" data buffers with !ddir, !lden, and !uden, and produces the ready
" signal for the SONIC with !sonicrdy. Wait states are inserted by
" memory (cdchrdy not asserted) and by the signal translator(SIG5) pal
" (synwait not asserted).
" declarations

"declarations

    TRUE,FALSE = 1,0;
    H,L = 1,0;
    x,z,c = .X,..Z,..C.;

    GND,VCC
        pin 12,24;

"outputs
        ddir,uden,lden,irq9,irq7,irq6,irq3,sonicrdy
        pin 16, 22, 21, 20, 19, 18, 17, 15;

"inputs
        pos1,pos2,swr,mwr,hlda,cmd,int,lcardsel,cdchrdy,synwait,ds,lchsetup, smack
        pin 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,11, 13, 14;

"equates
        sel = [pos2,pos1];

equations

        irq9 = 0;
        irq7 = 0;
        irq6 = 0;
        irq3 = 0;

enable irq9 = int & (sel == 3);
enable irq7 = int & (sel == 2);
enable irq6 = int & (sel == 1);
enable irq3 = int & (sel == 0);

!ddir = !swr & !hlda #           "Data buffer direction
        mwr & hlda;

!lden = hlda & !ds #           "Address and Lower Data buffer enable (D7-D0)
        !lchsetup #
        !lcardsel & !cmd;

!uden = hlda & !ds #
        !smack & !cmd;           "Upper data buffer enable (D16 - D31)

```

## PAL EQUATIONS (Continued)

```
!sonicrdy = cdchrdy & synwait;      "ready signal for the SONIC and sig.  
trans.
```

```
end irqsel;
```

TL/F/10748-28

## PAL EQUATIONS (Continued)

```

module arbmac;          flag '-r2','-t2';

title
'PAL20L8                National Semiconductor
MicroChannel Bus Arbiter State Machine          1/8/89
file name: ARBMAC.ABL                               wl'

ARBMAC device 'P20L8';

" PAL DESCRIPTION

" This pal controls the ARBVEC pal when it may drives the arbitration
" vector. This pal consists on an asynchronous state machine. These
" state machine equations (!q0 - !q3) are not reduced (ABEL™ will do this).
" The outputs are described as follows:
"   enarb = enables the ARBVEC pal to drive the arb. vector
"   hlda = hold acknowledge to the SONIC
"   burst = BURST signal on the microchannel bus (open collector
"           type output).
"   preout = PREMPT signal on the microchannel bus (open collector
"           type output).

"declarations

TRUE,FALSE = 1,0;
H,L = 1,0;
X,Z,C = .X.,.Z.,.C.;

GND,VCC
pin 12,24;

"outputs
q3,q2,q1,q0,burst,preout,hlda,enarb
pin 21,20,19,18,17,16,22,15;

"inputs
hold,arbqnt,buswin,fair,prein,chrst
pin 1,2,3,4,5,6;

" States of Arbiter

st = [q3,q2,q1,q0];

idle = st == [1,1,1,1];
req = st == [1,1,1,0]; "request uchannel bus; preout (q0) active
arb = st == [1,0,1,0]; "vectoring arb priority; preout(q0), enarb
active
lose = st == [1,0,0,0]; "lost arb battle; preout (q0) active
xfer1 = st == [0,0,1,0]; "intermdiate state to xfer2
xfer2 = st == [0,0,1,1]; "xfering data on bus; burst, hlda active
xfer3 = st == [1,0,1,1]; "intermediate state to idle
pen = st == [0,1,1,1]; "holding pen when fairness is enabled
esc1 = st == [1,0,0,1]; "intermediate state to esc2
esc2 = st == [1,1,0,1]; "intermediate state to idle

equations

```

## PAL EQUATIONS (Continued)

```

!q0 = idle & hold & !arbgnt & !chrst #   " This also -PREEMPT on the MCA bus
      req  & hold & !arbgnt & !chrst #
      req  & hold & arbgnt & !chrst #
      arb  & arbgnt & !chrst #
      arb  & !arbgnt & buswin & !chrst #
      arb  & !arbgnt & !buswin & !chrst #
      lose & hold & !arbgnt & !chrst #
      lose & hold & arbgnt & !chrst;

!q1 = arb  & !arbgnt & buswin & !chrst #
      lose & hold & !arbgnt & !chrst #
      lose & !hold & !chrst #
      escl & !chrst;

!q2 = req  & hold & arbgnt & !chrst #
      arb  & arbgnt & !chrst #
      arb  & !arbgnt & buswin & !chrst #
      arb  & !arbgnt & !buswin & !chrst #
      lose & hold & !arbgnt & !chrst #
      lose & hold & arbgnt & !chrst #
      lose & !hold & !chrst #
      xfer1 & !chrst #
      xfer2 & hold & !arbgnt & !chrst #
      xfer2 & !hold & !arbgnt & !fair & !chrst;

!q3 = arb  & !arbgnt & !buswin & !chrst #
      xfer1 & !chrst #
      xfer2 & hold & !arbgnt & !chrst #
      xfer2 & !hold & !arbgnt & fair & !chrst #
      xfer2 & arbgnt & !chrst #
      pen  & !prein & !chrst;

enarb  = arb # xfer1 # xfer2; "enables arb vector (ARBO - 3) on bus
hlda   = xfer2;              "HOLD ACK to SONIC
enable burst  = xfer2;      "-BURST on MCA bus (tri-state output)
enable preout = !q0;        "-PREEMPT on MCA bus (tri-state output)

burst  = 0;
preout = 0;

end arbm2;

```

TL/F/10748-31

## PAL EQUATIONS (Continued)

```

module arb;          flag '-r2','-t2';

title
  'PAL20L8                      National Semiconductor
  MicroChannel Bus Arbiter Vector 1/9/90
  ARBVEC                      wl'

ARBVEC device 'P20L8';

" PAL DESCRIPTION

" This pal dumps the arbitration vector onto the MicroChannel bus
" (ARB0 - 3) when the ARBMAC pal enables it (by ENARB). ARB3 - 0
" are open collector type outputs (only driven low). This pal
" indicates it has won the bus by driving -BUSWIN low.

"declarations

  TRUE,FALSE = 1,0;
  H,L = 1,0;
  X,Z,C = .X.,.Z.,.C.;

  GND,VCC
    pin 12,24;

" Outputs
  BUSWIN,ARB3,ARB2,ARB1,ARB0,Q,Q2,_RST
  pin 22,21,20,19,18,17,16,15;

" Inputs
  POS15,POS14,POS13,POS12,HOLD,ENARB,CHRST,Q3
  pin 1,2,3,4,5,6,7,8;

equations

  ARB3 = 0;          "When enabled, these tri-state outputs
  ARB2 = 0;          "will pull these arbitration lines low.
  ARB1 = 0;
  ARB0 = 0;

  Q = (ARB3 # !POS15);
  Q2 = (ARB3 # !POS15) & (ARB2 # !POS14);

!BUSWIN = Q2 & Q3 & (ARB0 # !POS12) & ENARB & HOLD;

enable ARB3 = !POS15 & ENARB & HOLD;
enable ARB2 = !POS14 & Q & ENARB & HOLD;
enable ARB1 = !POS13 & Q2 & ENARB & HOLD;
enable ARB0 = !POS12 & Q2 & Q3 & ENARB & HOLD;

_RST = !CHRST;      " reset for the adapter

end arb;

```



## PAL EQUATIONS (Continued)

```
module sig;          flag '-r2','-t2';
```

```
title
```

```
'PAL16R4                      National Semiconductor
MicroChannel Signal Translator for the SONIC          1/1/90
file name: SIG.ABL                                wl'
```

```
SIG device 'P16R4';
```

```
" PAL DESCRIPTION
```

```
" This pal provides the signal conversion from the SONIC to the
" MicroChannel bus. The state machine used is written in the
" ABEL™ design format.
```

```
"declarations
```

```
TRUE,FALSE = 1,0;
H,L = 1,0;
x,z,c = .X.,.Z.,.C.;
```

```
GND,VCC
    pin 10,20;
```

```
"outputs
q3,q2,q1,q0,s0,s1,cmd,delayads
    pin 17,16,15,14,19,18,13,12;
"q0 is used for -ADL on MCA bus
```

```
"inputs
bsck,ads,mwr,hlda,sonicrdy,fast,enb,ds
    pin 1,2,3,4,5,6,11,7;
```

```
input = [ads,hlda,sonicrdy,fast];
```

```
" States of Translator
```

```
idle   = ^b1111;
data   = ^b1011;
wait   = ^b1101; "wait state to delay assertion of -ADL
addlcht = ^b1110; "asserting -ADL on MCA bus
cmd1   = ^b0101; "beginning to assert -CMD;
cmd2   = ^b0011; "still asserting -CMD; return to DATA state
        " on next clock
cmd3   = ^b0001; "still asserting -CMD; inserting 1st wait-state
        "for MCA's synchronous mode
```

```
data_st = [q3,q2,q1,q0] == [1,0,1,1];
wait_st = [q3,q2,q1,q0] == [1,1,0,1];
addl_st = [q3,q2,q1,q0] == [1,1,1,0];
cmd1_st = [q3,q2,q1,q0] == [0,1,0,1];
cmd3_st = [q3,q2,q1,q0] == [0,0,0,1];
idle_st = [q3,q2,q1,q0] == [1,1,1,1];
```

TL/F/10748-33

1

## PAL EQUATIONS (Continued)

```

state_diagram [q3,q2,q1,q0]

State idle: case (input == [0,1,x,0]) :wait;
              (input == [1,1,x,x]) :idle;
              (input == [x,0,x,x]) :idle;
              (input == [0,1,x,1]) :addlcht; "fast mode
        endcase;

State data: case (input == [1,1,x,x]) :data;
              (input == [0,1,x,0]) :wait;
              (input == [x,0,x,x]) :idle;
              (input == [0,1,x,1]) :addlcht; "fast mode
        endcase;

State wait: case (input == [x,1,x,x]) :addlcht;
              (input == [x,0,x,x]) :idle;
        endcase;

State addlcht: case (input == [x,1,0,x]) :cmd2;
              (input == [x,1,1,x]) :cmd1;
              (input == [x,0,x,x]) :idle;
        endcase;

State cmd1: case (input == [x,1,x,x]) :cmd3;
              (input == [x,0,x,x]) :idle;
        endcase;

State cmd2: case (input == [x,1,x,x]) :data;
              (input == [x,0,x,x]) :idle;
        endcase;

State cmd3: case (input == [x,1,x,x]) :data;
              (input == [x,0,x,x]) :idle;
        endcase;

equations

enable s0 = hlda;
enable s1 = hlda;
enable cmd = hlda;

" MCA Signals

!s0 = mwr & !data_st & !idle_st; "-S0 for MCA bus

!s1 = !mwr & !data_st & !idle_st; "-S1 on MCA bus

!cmd = !q3 & q2 & delayads & !ds & hlda #
      !q3 & !q2 & delayads & !ds & hlda # "-CMD for MCA bus
      q3 & !q2 & delayads & !ds & hlda;

delayads = ads; "delaying ADS for the -CMD term to
                "prevents glitches from occurring
                "during the transitions from the DATA
                "to the WAIT state

end sig;

```

## PAL EQUATIONS (Continued)

```

module logic;

title
'PAL16R4 MicroChannel Logic                               National Semiconductor
file name: logic.abl                                     1/9/90'

LOGIC device 'P16R4';

"declarations

    TRUE, FALSE = 1,0;
    H,L = 1,0;
    X,Z,C = .X.,.Z.,.C.;

    GND, VCC
        PIN 10, 20;

"inputs
    bsck,hldaout,_adl,arb1,pos13,enb
    pin 1,2,3,4,5,11;
"outputs
    q3,adl,hlda,_hlda,hlda1
    pin 12,13,15,14,19;

equations

    hlda := hldaout;
    _hlda := !hldaout;
    hlda1 = hlda;
    adl = !_adl;
    q3 = arb1 # !pos13;

test_vectors ([bsck,hldaout,_adl,arb1,pos13,enb] -> [hlda,_hlda,hlda1,adl,q3])
[C,1,1,0,0,0] -> [1,0,1,0,1];
[C,0,0,0,1,0] -> [0,1,0,1,0];
[C,1,X,1,1,0] -> [1,0,1,X,1];
end log;

```

TL/F/10748-35

# DP83932 SONIC™ Bus Operations Guide

National Semiconductor  
Application Note 745  
Wesley Lee



This application note is intended to be a supplementary document for the DP83932 SONIC datasheet, expanding upon the bus functional descriptions found in the datasheet. It is recommended that you are familiar with the bus operations of the SONIC before reading this document.

This application note gives additional examples of the SONIC's bus operations to illustrate a broader picture of receptions, transmissions, etc. Where possible, special conditions are included to show all conceivable bus operations performed by the SONIC. Detailed figures are shown to enhance clarity. This document is divided into two sections for bus master and slave operations. The bus master section details bus operations during transmission, receptions and load CAM operations, and the slave access section describes SONIC register accesses during idle and non-idle conditions.

## TERMS AND ABBREVIATIONS

In this document certain terms and abbreviations will be used to describe the bus operations of the SONIC. These words are defined as follows:

**Block Transfer:** A multiple transfer bus operation in which the address increments for each transfer.

**Word:** Refers to a 16-bit quantity; a double word is a 32-bit quantity.

**Memory Cycle:** The basic cycle which the SONIC reads from or writes to memory.

**Bus Tenure:** The complete time the SONIC uses the bus during a block transfer.

**Bus Latency:** This is the time from when the SONIC requests the bus to when the SONIC is granted the bus.

**CAM:** Content Addressable Memory

**TDA:** Transmit Descriptor Area

**TBA:** Transmit Buffer Area

**RRR:** Receive Resource Area

**RDA:** Receive Descriptor Area

**RBA:** Receive Buffer Area

**CDA:** CAM Descriptor Area

## 1.0 BUS MASTER OPERATIONS

### 1.1 The Basic Block Transfer Cycle

The basic transfer cycle of the SONIC is composed of three basic operations: (1) acquiring the bus, (2) transferring data onto/from the bus, and (3) relinquishing the bus. Operations (1) and (3) are described in detail in Section 5.4 of the DP83932 datasheet or Section 7.3 of the DP83934 datasheet and will not be discussed here. Operation (2), however, will be more fully explained.

When the SONIC uses the bus, it transfers data to/from one specific area in memory (i.e., RBA, RDA, RRA, TDA, or TBA) as indicated by the bus status pins  $S<2:0>$ . If the SONIC needs to transfer the data to multiple areas in memory, it deasserts its bus request (HOLD or  $\overline{BGACK}$ ), then requests the bus again (HOLD or  $\overline{BR}$ ). During its tenure on the bus, the SONIC transfers a programmed number of words to memory, depending on where data is placed. The number of transfers to the descriptor areas (TDA, RDA, RRA, and CDA), are shown in the following table. Note that (during SONIC descriptor access) since the upper word ( $D<31:16>$ ) is not used in 32-bit mode, the number of transfers are the same for both 16-bit and 32-bit modes.

TABLE 1-1. Number of Memory Transfers to the Descriptor Areas

Area	Number	R/W	When
CDA	4	R	All bus tenures except the last one
	5	R	Last bus tenure. The additional access is to load the CAM Enable register.
TDA	6	R	First descriptor fetch
	3	R	Additional fragment pointer and size fetches, if any
	2	R/W	Status and link access
RDA	7	R/W	Updating receive descriptor information
	6	R/W	Updating receive descriptor information but SONIC has read EOL = 1
	2	R/W	Re-reading RXpkt.link and writing to RXpkt.in_use when EOL has previously been detected as 1. The SONIC writes to the in_use field when EOL now reads 0.
	1	R	Re-reading the RXpkt.link as above but the SONIC still reads EOL = 1.
RRR	4	R	All bus tenures

For buffer area transfers (TBA and RBA), the number of memory transfers is determined by the FIFO threshold and whether the SONIC is in "empty/fill" or "exact block" transfer modes, programmed in the Data Configuration register. For "exact block" transfer mode, the SONIC transfers the same number of words (or double words) as are programmed for the FIFO threshold. For example, if you programmed 4 double words as the threshold for the receive FIFO, the SONIC will transfer this amount of data to memory per bus tenure. There are two exceptions to this rule, however. First, during transmission or reception, if the packet is not a multiple of the FIFO threshold, the last bus tenure will contain less transfers than the FIFO threshold. Second, for high transmit FIFO thresholds (12 words or 14 words), the SONIC will fill the transmit FIFO only as much as needed to completely fill it (and not overflow it). Thus, if you choose a 12 word transmit FIFO threshold, the first bus tenure will transfer 12 words, but the second tenure will only transfer 4 words (12 words + 4 words = 32 bytes). This last example assumes that the bus latencies are zero.

For "empty/fill" mode, the number of transfers is also dependent on the bus latency. When the FIFO threshold has been reached, the SONIC will either completely empty the FIFO during reception or completely fill the FIFO during transmission. At the time of the bus request, the FIFO threshold equals the number of words in the FIFO, but due to bus latencies, additional bytes may have entered the FIFO (during reception). Thus, the number of words transferred during a bus tenure in this mode is the FIFO threshold plus the additional bytes that have entered the FIFO during reception or minus the bytes that have been serialized during transmission.

**1.2 Packet Reception**

This section gives a step-by-step description of the SONIC receiving a 68-byte packet. The initial conditions are shown below.

Initial conditions:

- (a) The incoming packet is one that the SONIC will accept.
- (b) The SONIC has detected that EOL = 1 from the previous reception, but the software has subsequently appended another receive descriptor before receiving this packet.
- (c) The packet begins on a double word boundary.
- (d) The Data Configuration register has been configured for:
  - 32-bit data width mode (DB5 = DW = 1)

- 4 double word Receive FIFO threshold (DB3,2 = RF1,0 = 1,0)
  - Exact Block Transfer mode (DB4 = BMS = 1)
- (e) The packet has crossed over the End of Buffer Count (EOBC) register during this reception; hence the SONIC will need to use another Receive Buffer Area (RBA).

The reception is described as follows.

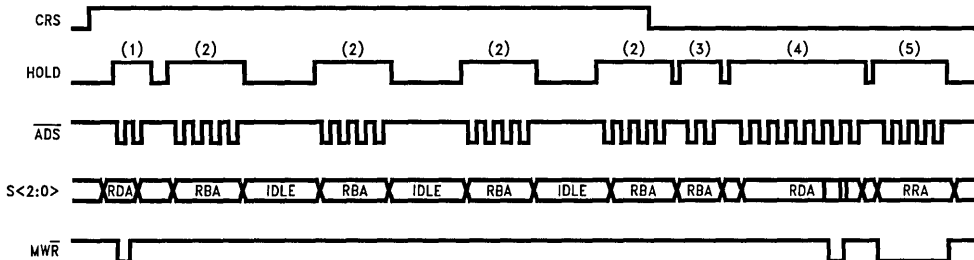
**Note:** The numbers in this section correspond to the numbers in *Figure 1-1*.

- (1) Because of condition (a), the SONIC reads the RXpkt.link again to see if the software has subsequently reset the EOL bit to zero. Since it has (condition [b]), the SONIC writes to the RXpkt.in\_use field and buffers the packet to the RBA. Note that this step is skipped if the SONIC has sufficient descriptors.
- (2) Once the receive FIFO has reached its threshold (4 double words), the SONIC will write 4 double words during its bus tenure. For a 68-byte packet, the SONIC will perform this operation 4 times.
- (3) During the last RBA bus tenure, the packet has ended (CRS goes low). The SONIC requests the bus once again (in 3 bus clocks) and flushes the remaining bytes in the FIFO (8 bytes). The first 4 bytes are the remainder of the packet and the last 4 bytes are the receive status that is automatically written into the FIFO by the SONIC. These last 4 bytes are extraneous to the RBA and are overwritten during the next reception. The usable receive status is written to the Receive Descriptor Area.

**Note 1:** If the packet size is not a multiple of the memory transfer size (16 bits or 32 bits), the SONIC will pad the last memory transfer with 1's as necessary.

**Note 2:** If any of the last 4 bytes exceeds the length of the Receive Buffer Area, these bytes will not be written to memory.

- (4) The SONIC writes the status information in the Receive Descriptor Area. The SONIC performs 7 consecutive memory transfers during its bus tenure (5 writes to the RXpkt.status, RXpkt.byte\_count, RXpkt.pkt\_ptr0, RXpkt.pkt\_ptr1, and RXpkt.seq\_no. fields, 1 read to the RXpkt.link field, and 1 write to the RXpkt.in\_use field). See *Figure 1-3* and Section 1.2.2 for further details.
- (5) Because of condition (e), the SONIC requests the bus again (in 3 bus clocks) and fetches a resource descriptor from the Receive Resource Area. The SONIC reads this area in 4 consecutive memory read operations.

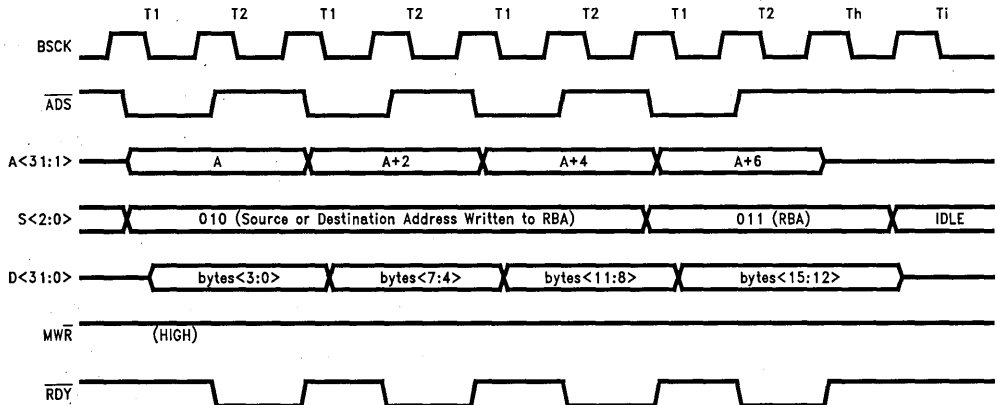


**FIGURE 1-1. Complete Reception of a 68-Byte Packet**

TL/F/11139-1

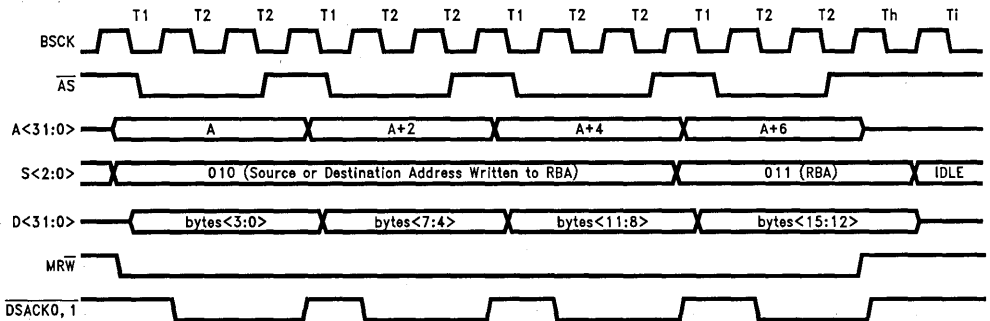
**1.2.1 Detail of Access to the RBA**

Figure 1-2a and 1-2b show the first SONIC access to the RBA for BMODE = 0 and 1. Note that the status pins S<2:0> change from 0,1,0 to 0,1,1 as the SONIC finishes writing the Source Address of the packet and continues buffering the rest of the packet.



**FIGURE 1-2a. First RBA Access for Storing Packet (BMODE = 0, Synchronous Mode)**

TL/F/11139-2

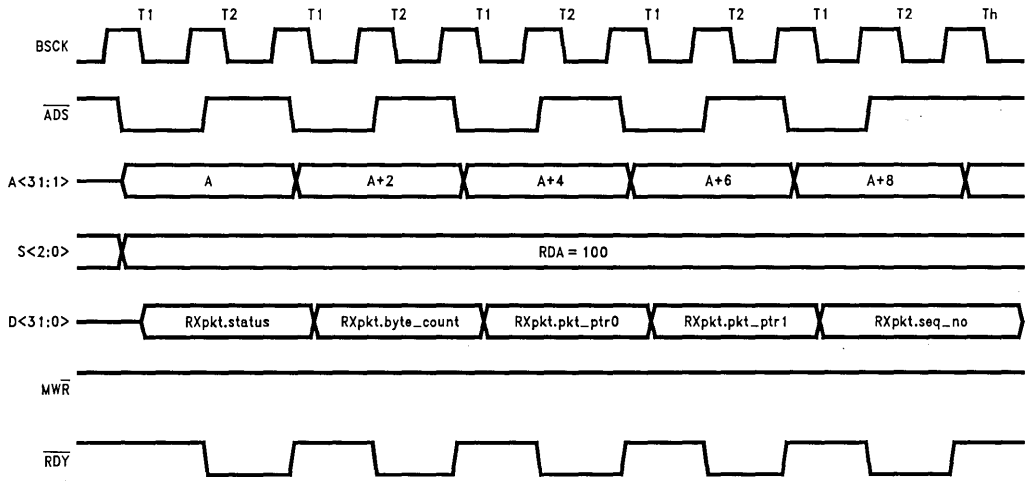


**FIGURE 1-2b. First RBA Access for Storing Packet (BMODE = 1, Asynchronous Mode)**

TL/F/11139-3

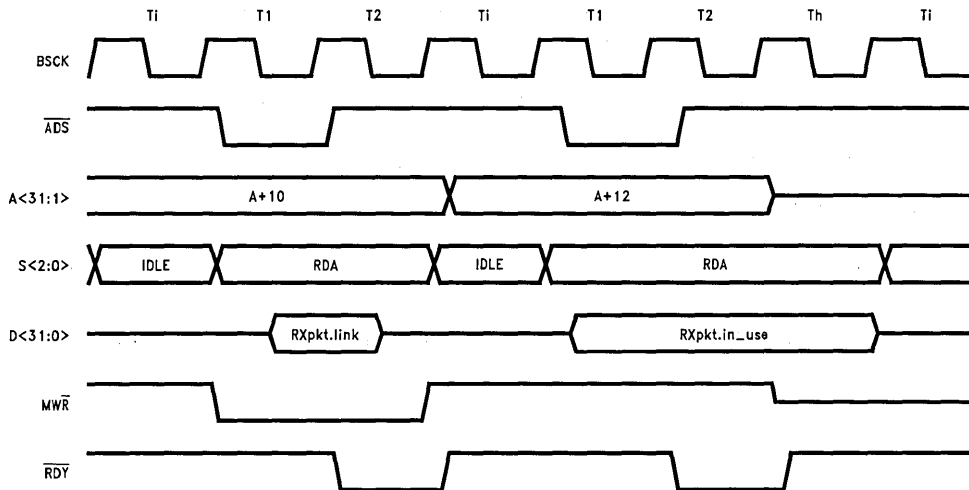
**1.2.2 Detail of Access to the RDA**

Figure 1-3a and 1-3b shows the SONIC accessing the RDA for both BMODE = 0 and 1. Note that this block transfer contains both read and write accesses. Also note that the status pins S<2:0> briefly change from RDA (1,0,0) to Idle (1,1,1) between the read and write operations. This occurs between the RXpkt.seq\_no and RXpkt.link accesses, and between the RXpkt.link and RXpkt.in\_use accesses and is accompanied by a Ti bus clock state.



TL/F/11139-4

**FIGURE 1-3a. RDA Access for Storing Descriptor Information (BMODE = 0, Synchronous Mode)**



TL/F/11139-5

**FIGURE 1-3a. RDA Access for Storing Descriptor Information (BMODE = 0) (Continued)**

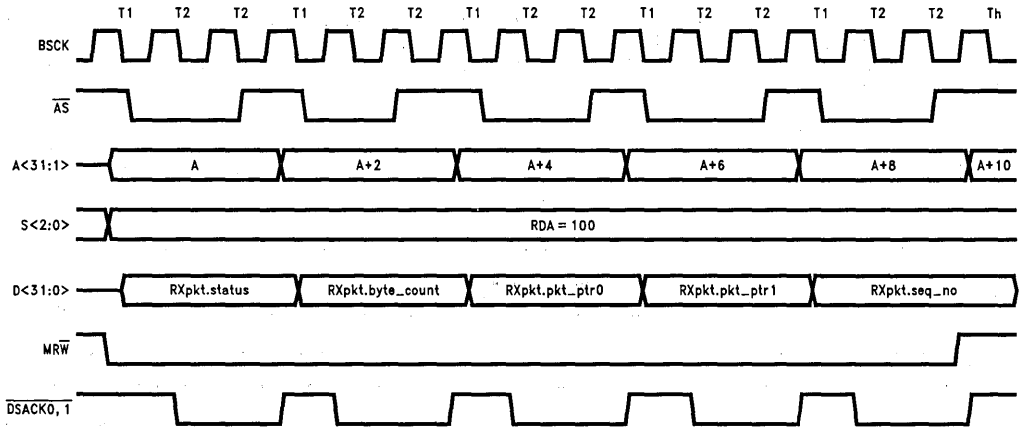


FIGURE 1-3b. RDA Access for Storing Descriptor Information (BMODE = 1, Asynchronous Mode)

TL/F/11139-6

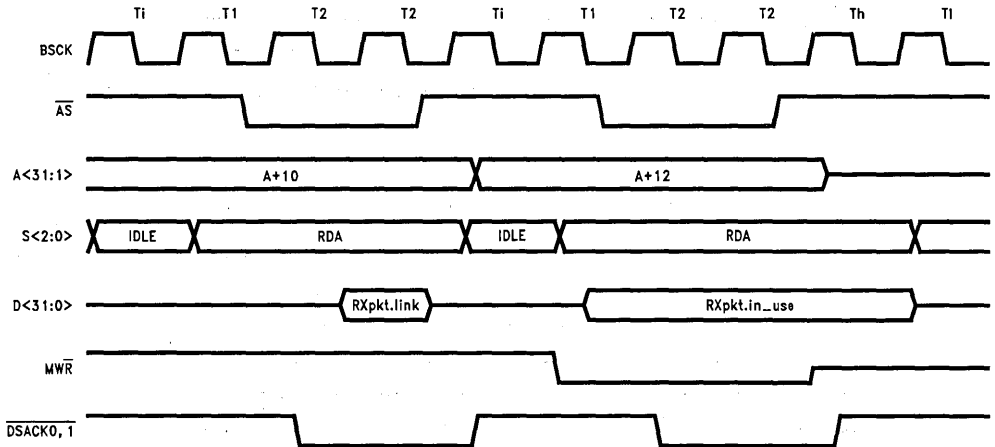


FIGURE 1-3b. RDA Access for Storing Descriptor Information (BMODE = 1) (Continued)

TL/F/11139-7



### 1.3 Packet Transmission

This section gives a step-by-step analysis of a complete transmission using the initial conditions below.

Initial conditions:

- (a) The Data Configuration register has been configured for:
  - 32-bit data wide mode (DB5 = DW = 1)
  - 16-byte Transmit FIFO threshold (BD1, 0 = TF1, 0 = 0, 1)
  - Exact Block Transfer mode (DB4 = BMS = 1)
- (b) The packet consists of 2 fragments. The first one is 48 bytes long and the last one is 20 bytes long.
- (c) Both fragments are double-word aligned.

The transmit operation is described as follows: (the numbers in this section correspond to the numbers in *Figure 1-4*)

- (1) Before the SONIC transmits, it fetches a descriptor from the Transmit Descriptor Area (TDA) to load its transmit registers. In 6 consecutive memory read operations, the SONIC reads the TXpkt.config, TXpkt.pkt\_size, TXpkt.frag\_count, TXpkt.frag\_ptr0, TXpkt.frag\_ptr1, and TXpkt.frag\_size fields. Note that the TXpkt.status field is skipped during the first TDA access. Note also that if a collision occurs, forcing the SONIC to retransmit, the SONIC will once again fetch the descriptor from the beginning (i.e., starting at TXpkt.config).
- (2) After fetching the descriptor, the SONIC begins loading the FIFO to its transmit threshold. The SONIC performs 4 consecutive memory operations in the Transmit Buffer Area (TBA) per bus tenure. Note that the fragment may begin on any byte boundary; if this is the case, the SONIC reads the corresponding double word which contains the beginning of the packet.
- (3) The SONIC immediately requests for the bus again (in 3 bus clocks) because the number of words in the FIFO is equal to or less than the Transmit FIFO threshold. When the threshold has been exceeded, the SONIC commences transmission (TXE goes high). Subsequent re-

quests for the TBA will not occur until the serializer has removed enough bytes from the FIFO to lower it below its threshold.

- (4) Because of condition (b), the SONIC goes back to the Transmit Descriptor Area to obtain the pointer and length count of the next fragment. In three consecutive read operations, the SONIC will read the next TXpkt.frag\_ptr0, TXpkt.frag\_ptr1, and TXpkt.frag\_size fields.
- (5) At the end of transmission, the SONIC will write the status of the TXpkt.status field, then read the TXpkt.link field to locate the next descriptor.

#### 1.3.1 Detail of Access to the TDA

*Figure 1-5a* and *1-5b* shows the SONIC accessing the TDA at the end of transmission. The SONIC writes the status information at the beginning of the descriptor and reads the link field at the end of descriptor. (Note  $n$  = the number of fragments.) Since this access involves both a write and a read, there is a transition from TDA to Idle to TDA on the status lines in between writing the status and reading the link field. This transition is accompanied by a Ti bus clock state.

#### 1.3.2 Detail of Access to the TBA

*Figure 1-6a* and *1-6b* shows the SONIC accessing the TBA when the transmit FIFO has been programmed for (1) 32-bit mode (2) exact block transfer mode, and (3) a 4 double word threshold.

#### 1.4 Loading the CAM (Content Addressable Memory)

After the CAM descriptor Area has been initialized and the Load CAM command issued to the SONIC, the SONIC will read the CAM Descriptor Area (CDA) and load its CAM. The SONIC, in 4 memory read cycles, accesses memory and loads one CAM entry per bus tenure. During the last block transfer, the SONIC reads one additional word to load its CAM Enable register. In the example illustrated in *Figure 1-7*, the SONIC has been programmed to load 4 CAM locations.

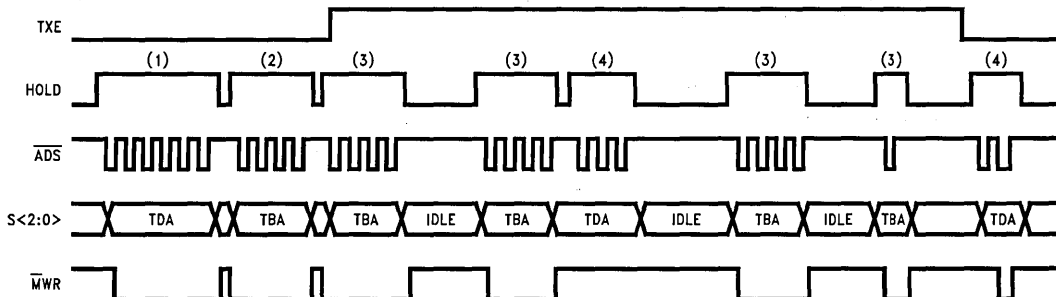


FIGURE 1-4. Complete Transmission of a 68-Byte Packet

TL/F/11139-8

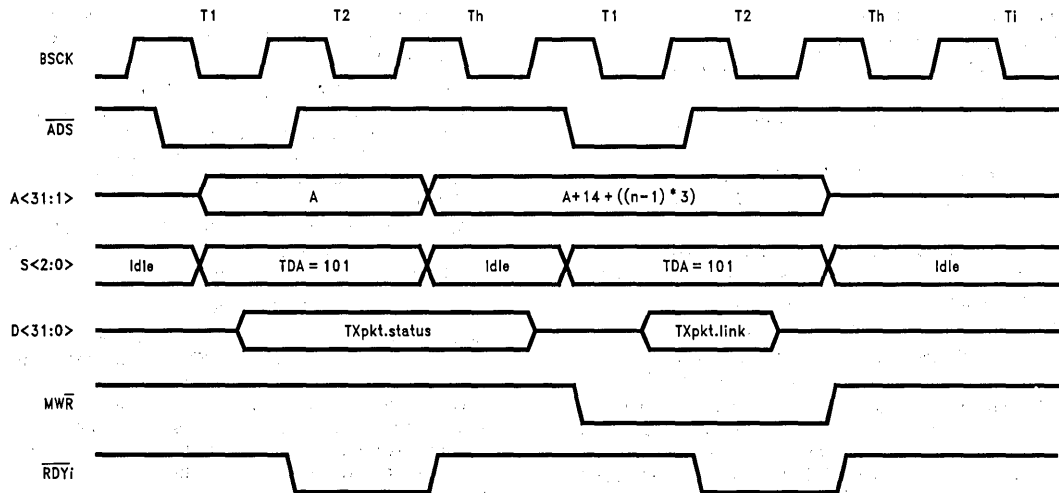


FIGURE 1-5a. Last TDA Access (BMODE = 0, Synchronous Mode)

TL/F/11139-9

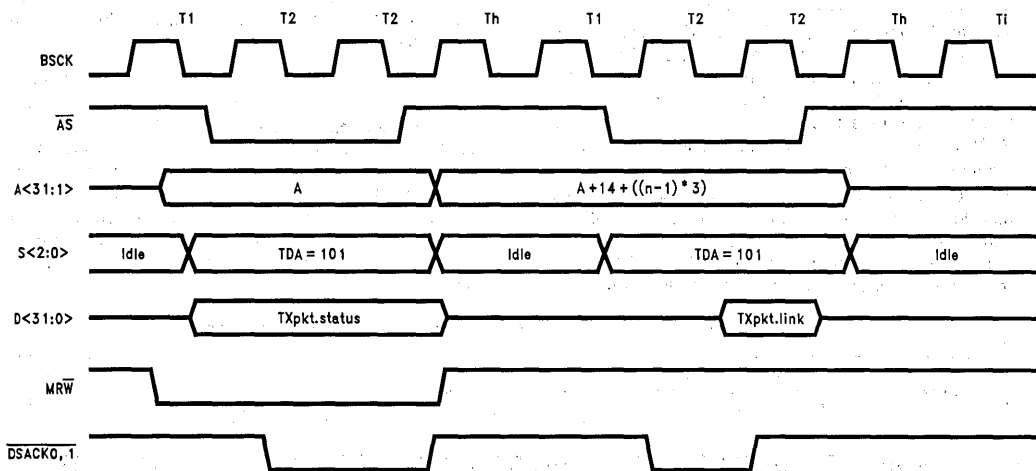


FIGURE 1-5b. Last TDA Access (BMODE = 1, Asynchronous Mode)

TL/F/11139-10

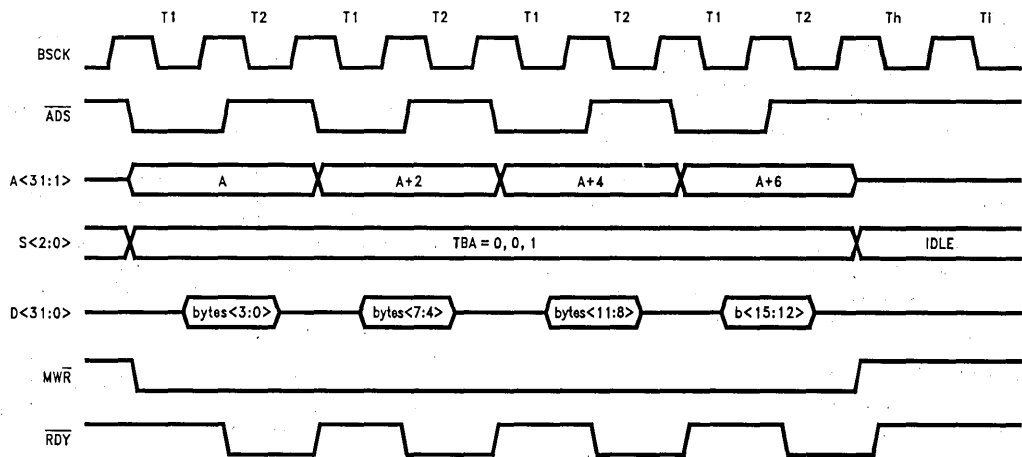


FIGURE 1-6a. Typical TBA Access (BMODE = 0, Synchronous Mode)

TL/F/11139-11

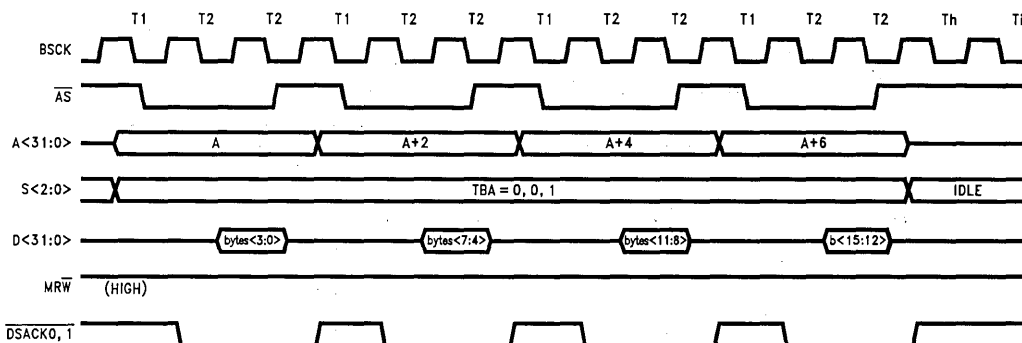


FIGURE 1-6b. Typical TBA Access (BMODE = 1, Asynchronous Mode)

TL/F/11139-12

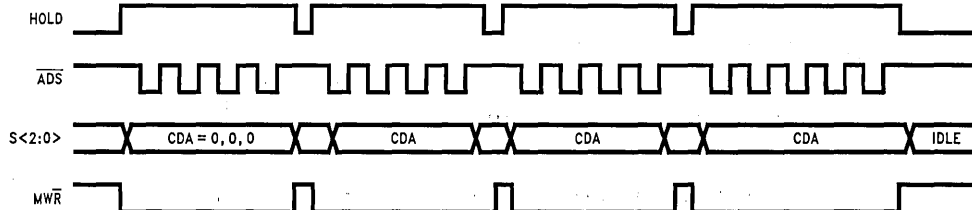


FIGURE 1-7. Updating 4 CAM Entries with the Load CAM Command

TL/F/11139-13

**2.0 SLAVE OPERATIONS**

The slave operations of the SONIC can be classified into two cases, (1) register access while the SONIC is idle, and (2) register accesses while the SONIC is not idle. The first case always occurs in single bus systems where the CPU and SONIC reside on the same bus. Only one bus master is allowed on the bus in such a system. The second case may occur in dual bus systems where the CPU and SONIC lie on different busses. In this case, the CPU may access the SONIC while it is currently using the bus (such as during transmission or reception). The SONIC does not respond immediately in this case, but finishes off its current bus master operation before responding to the register access. The following two sections give a step-by-step description of the slave operations.

**2.1 Register Access During Idle**

(Refer to Figures 2-1a and 2-1b)

(1) The CPU presents the register address, address strobe, chip select, and slave write/read strobe to the SONIC.

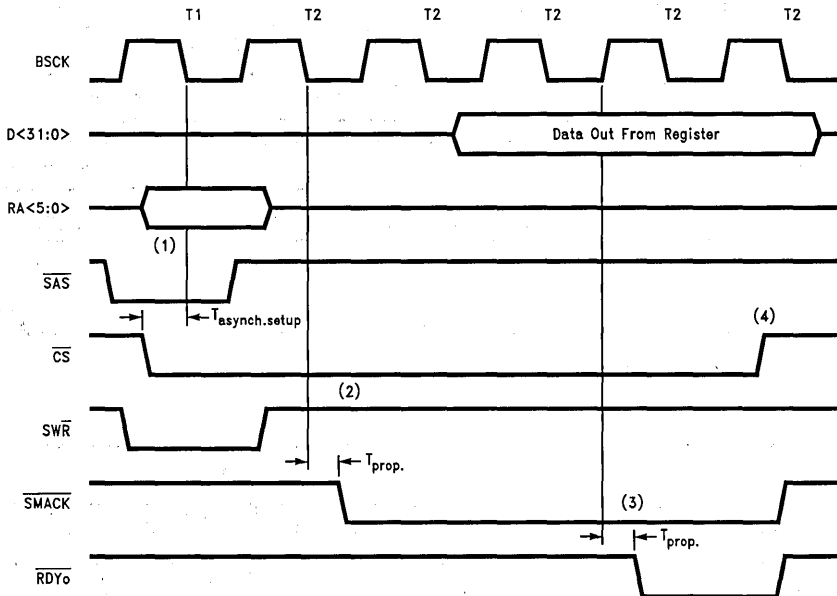
**Note:** For BMODE = 0, SAS must be asserted low before or at the same time that CS is asserted. The rising edge of SAS latches the register address, RA<5:0>, and slave direction strobe, SWR.

(2) The SONIC synchronizes chip select to the falling edge of bus clock and responds with slave and memory acknowledge (SMACK) at the next falling edge of bus clock.

**Note:** BMODE = 0, if SAS remains asserted (low) beyond the falling edge of CS, SMACK will not be asserted until SAS is deasserted high.

(3) For BMODE = 1, DSACK0,1 is generated 2 bus clocks after SMACK is asserted, and for BMODE = 0, RDY0 is generated 2.5 bus clocks after SMACK. These outputs indicate to the CPU that the slave access is over. For read cycles, register data is valid at the falling edge of the ready signal (RDY0 or DSACK0,1); for write cycles, the SONIC has latched the register data at the falling edge of the ready signal.

(4) The CPU completes the slave cycle by deasserting CS or reasserting SAS low for BMODE = 0, or deasserting CS or SAS for BMODE = 1. The earliest of these signals will terminate the slave cycle.



**FIGURE 2-1a. Register Read While SONIC is Idle (BMODE = 0)**

TL/F/11139-14

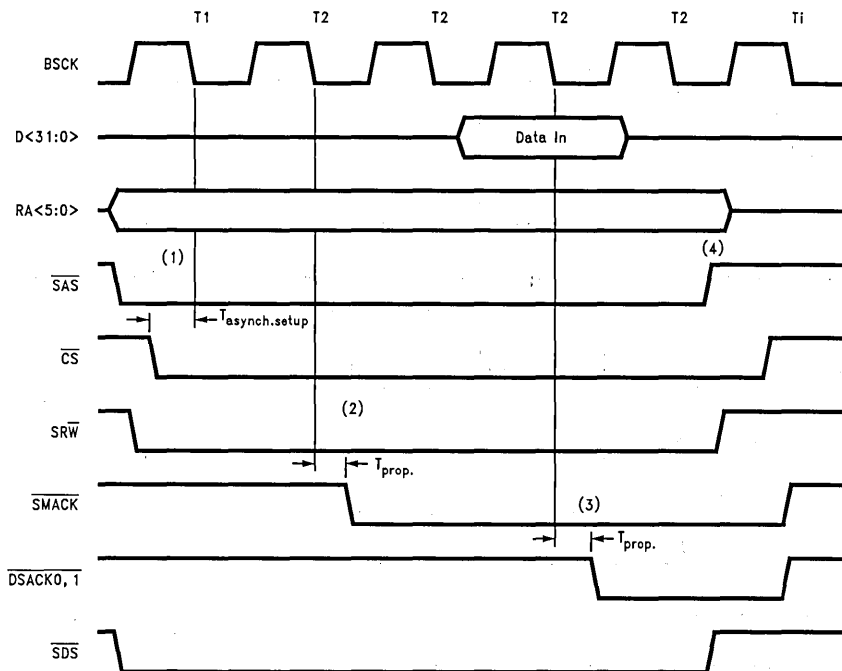


FIGURE 2-1b. Register Read While SONIC is Idle (BMODE = 1)

TL/F/11139-15

## 2.2 Register Access While SONIC is a Bus Master

Figures 2-2a and 2-2b show register accesses on the SONIC for both BMODE = 0 and BMODE = 1 respectively. These register accesses occur while the SONIC is a bus master (HOLD is high or BGACK is low). Notice how the bus states for the SONIC controlled DMA access (shown as states Ts1, Ts2 etc.) and the CPU controlled register access (shown as states Tc1, Tc2, etc.) overlap. They both start at the same time, but the register access does not complete until the SONIC DMA access completes and the SONIC gets off the bus. The following description refers to the numbers in Figures 2-2a and 2-2b.

- (1) In order to initiate a slave transfer, the SONIC must sample CS low. Also, it must sample SAS high for BMODE = 0 or SAS low for BMODE = 1. Even when the SONIC is doing master mode accesses on the bus, it monitors CS and SAS. If the above conditions are met, the SONIC finishes the current master mode bus cycle, and then TRI-STATES off the bus.
- (2) The SONIC asserts slave and memory acknowledge (SMACK) off the falling edge of the clock on the first Tc2 after the Tsh state.

- (3) For BMODE = 1, DSACK0,1 is generated 2 bus clocks after SMACK is asserted, and for BMODE = 0, RDY0 is generated 2.5 bus clocks after SMACK. These outputs indicate to the CPU that the slave access is over. For read cycles, register data is valid at the falling edge of the ready signal (RDY0 or DSACK0,1); for write cycles, the SONIC has latched the register data at the falling edge of the ready signal.
- (4) The CPU terminates the slave cycle by deasserting chip select. Alternatively, the slave cycle can be terminated by driving SAS low for BMODE = 0, or SAS high for BMODE = 1.
- (5) After the CPU has terminated the slave cycle, the SONIC continues its bus master operations from where it left off.

## 2.3 Asserting MREQ to Access Shared Memory

Asserting MREQ to the SONIC has nearly the same effect as asserting CS. SMACK is generated identically as before, but the ready signal (RDY0 or DSACK0,1) is not asserted. The ready signal must be asserted by the memory control logic. Also, when using MREQ, it is not necessary to assert SAS.

**Note:** Both MREQ and CS must not be asserted simultaneously. This will cause spurious accesses to the SONIC's registers. Note also that the SONIC requires a recovery time of 2 bus clocks between the deassertion edge of one signal to the assertion edge of the other.

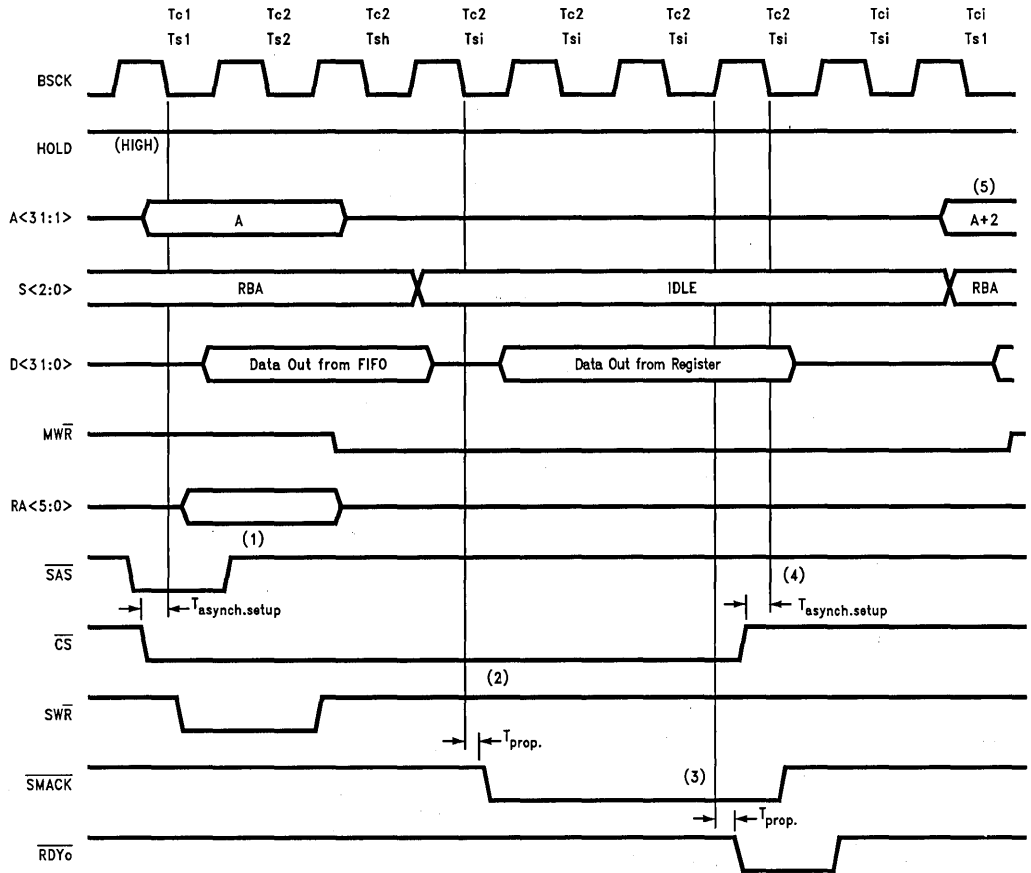


FIGURE 2-2a. Register Read While SONIC is Currently Using the Bus (BMODE = 0)

TL/F/11139-16

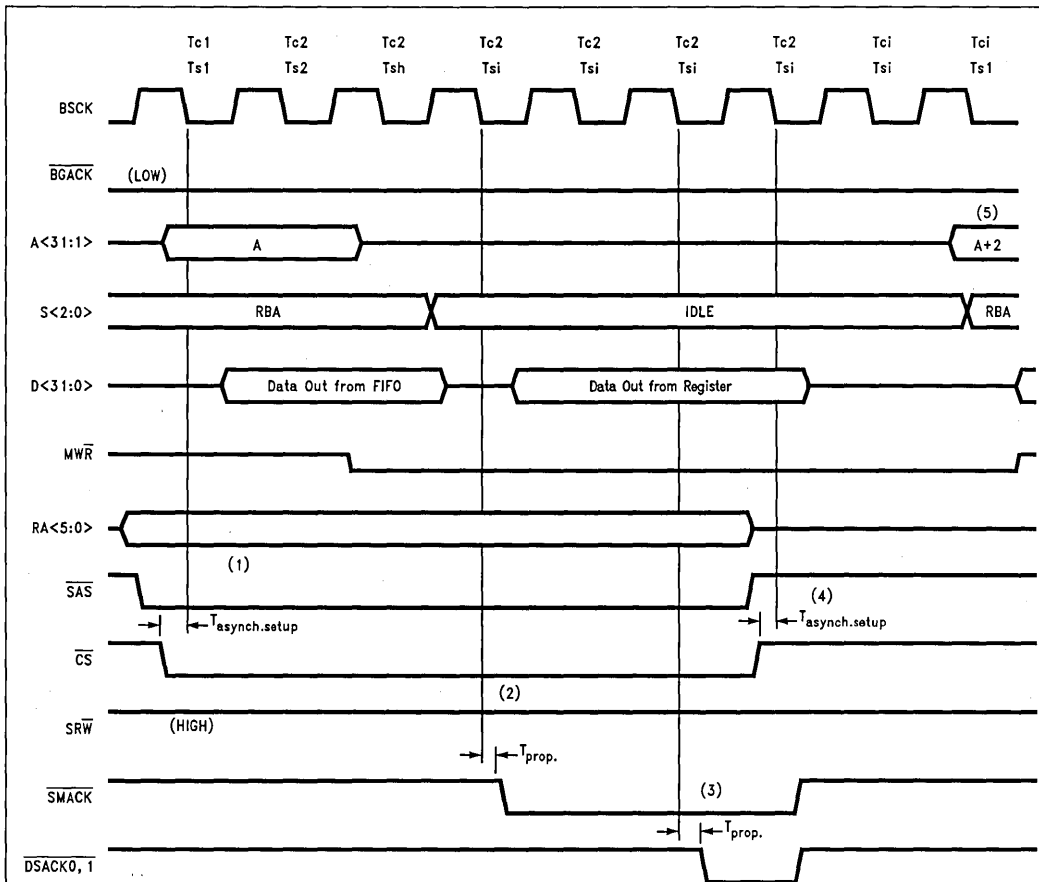


FIGURE 2-2b. Register Write While SONIC is Currently Using the Bus (BMODE = 1)

TL/F/11139-17

# Software Driver Programmer's Guide for the DP83932 SONIC™

National Semiconductor  
Application Note 746  
Wesley Lee  
Mike Lui



## INTRODUCTION

In the past, Ethernet chips have concentrated on interfacing well with the hardware, but have given the software interface only passing notice. While hardware designers may have been satisfied, the software developers were forced to write drivers for unwieldy silicon. Recently, with companies looking for ways to increase performance, they have found that the software interface is crucial and has been one of the bottlenecks in the system. A chip with an over constraining buffer management slows down the system by introducing more levels of indirection (pointers) than are truly needed by the system software. In view of these shortcomings, National surveyed a number of software developers to define a buffer management system which operates efficiently with the driver. Their basic response was *Keep it Simple*. The reasons were twofold. First, a simple software interface engenders a driver which is easy to write and secondly, a simpler, thus shorter, driver leads to a faster driver. The SONIC's buffer management epitomizes this with three salient features. First, only one level of indirection is used to reference data in memory; secondly, link-lists are chosen to endow the software developer with the flexibility to easily manipulate descriptors, and thirdly, a register-based command interface is provided to make commands fast and immediate.

## ABOUT THIS GUIDE

This guide will provide you the information needed to write a driver for the DP83932 System-Oriented Network Interface Controller (SONIC). You will first be introduced to basic algorithms using the SONIC's buffer management, then be shown actual implementation examples. It is recommended that you are familiar with the DP83932 SONIC datasheet before reading this document.

## 1.0 THE DRIVER SOFTWARE—SONIC INTERACTION

The key to making a Driver and all upper levels of the network software efficient, is to ensure that they must be capable of referencing received or transmitted packets via pointers and then conveying these pointers up to the next level of software. By employing pointers in this manner, needless packet copying from one area in memory to another is eliminated. As shown in *Figure 1-1*, the SONIC's descriptor areas, the RDA and TDA reference the received and transmitted packet and the RRA references the buffers for the received packets. The actual received and transmitted packets remain in their original locations in the RBA and TBA and are not copied elsewhere. In this section the basic algorithms are given to illustrate the usage of the RDA, RRA and TDA. Section 4.0 describes the implementation examples.

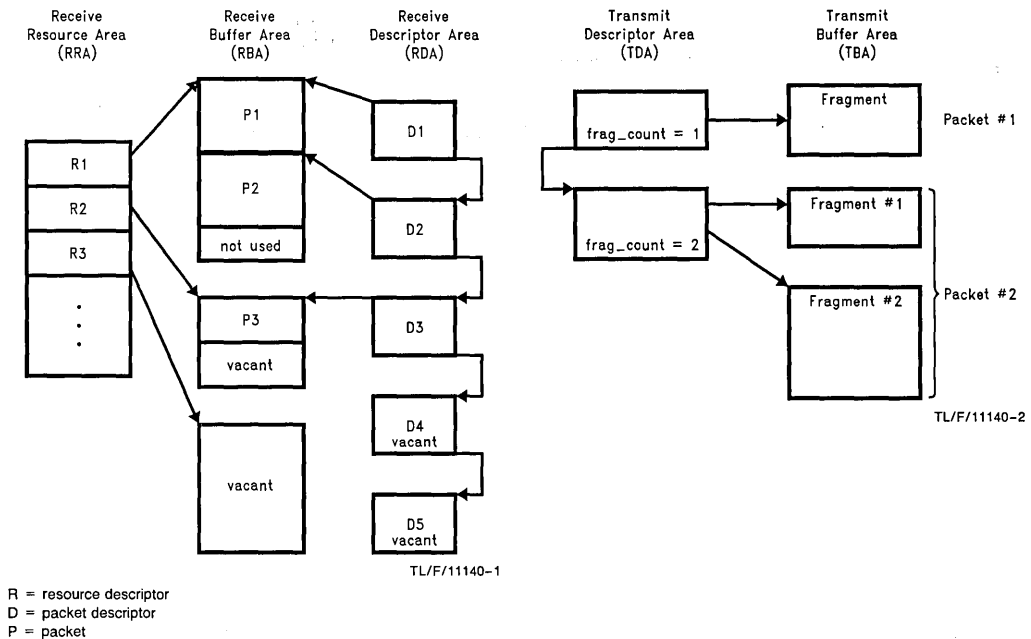


FIGURE 1-1. Overview of the SONIC's Buffer Management



### 1.1 Processing Packets in the Receive Descriptor Area (RDA)

After the SONIC has received the packet, it places the packet in the RBA and the packet information in the RDA. The Driver, in turn, processes this packet by locating the packet from the packet pointer (RXpkt.ptr0,1) fields in the RDA and then delivering the pointer up to the next level software for further processing. The Driver then returns the descriptor to the front of the list for reuse. This process is illustrated in *Figures 1-2 (a), (b), and (c)*. Note that the link-list allows descriptors to be appended to the front of the list in any order. Note also that no special considerations are required to append receive descriptors. The pseudo code below illustrates the simplicity in appending descriptors.

```
append_desor( )
    new_RXpkt.link = 1;
    old_RXpkt.link = new_RXpkt.status
    /* Old link field points to
    address of new status field */
```

### 1.2 Recycling Buffers in the Receiver Resource Area (RRA)

Intermixed with processing of packets, the Driver must also replenish the receive buffer pool by adding resources descriptors to the RRA. A suggested method for replenishing receive buffers is given in the following example. (This method assumes that more than one packet is stored in an RBA.)

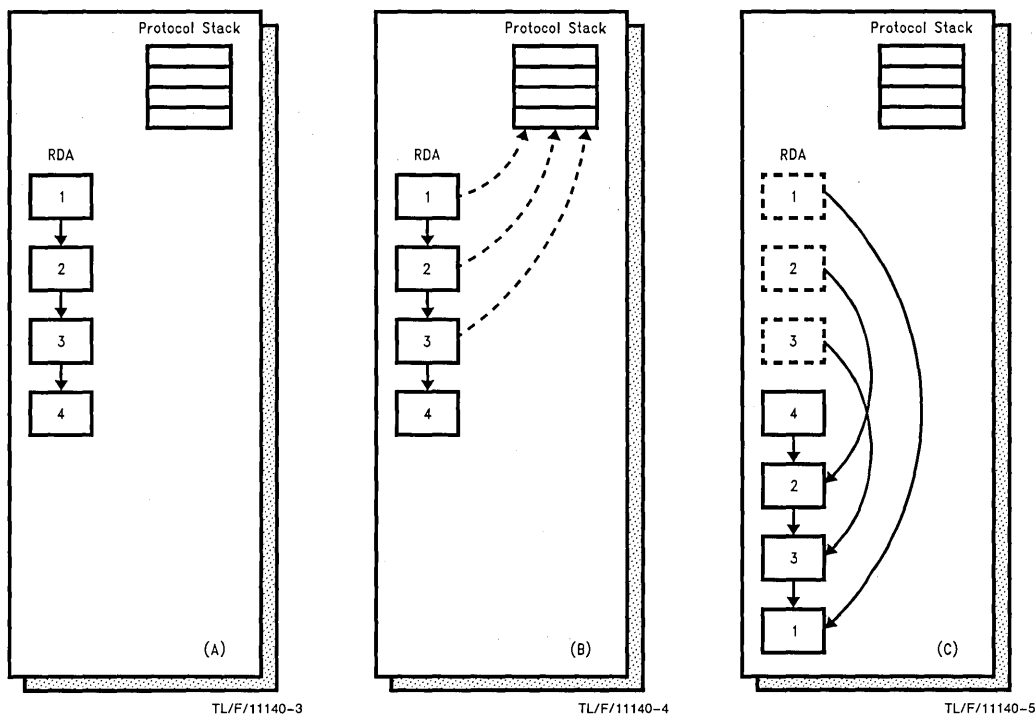


FIGURE 1-2 a, b, c. Processing Descriptors in the RDA

- (a) Initial condition: four descriptors are available for use  
 (b) Packets received: three packets having been received are then passed up to the upper level software for further processing.  
 (c) Packets processed: the upper level software having finished processing the packets, the Driver returns the descriptors to the front of the list.

The Driver allocates a fixed number of receive buffers (RBAs), determined at initialization, and recycles them as they are used. When the upper level software receives a packet from the driver (via pointers), it processes the packet at the location received (in the RBA), and when done, notifies the Driver of the freed memory space. The Driver, then, records this event by tallying the packet in a "scoreboard" corresponding to the RBA (see *Figure 1-3*). When the number of packets processed equals the total number of packets in an RBA, then RBA is free and may be returned to the RRA ring.

RBA #	Packets Processed	Total Packets In RBA
0	5	5
1	2	4
2	3	6
3	3	Unknown

FIGURE 1-3: RBA Scoreboard Example

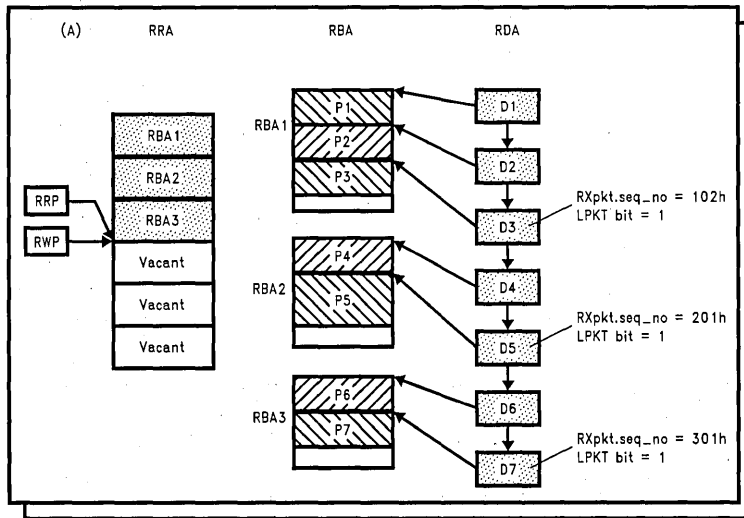
RBA #0 is now free since packets processed equals total packets in RBA. For RBA #3, the software does not yet know how many packets reside in an RBA since the SONIC has not finished using this RBA. When the software detects the LPKT bit set, the packet sequence number reveals the total number of packets (see below).

Because packets may be processed in any order (thus, packets may be freed up out of sequence), freeing up an RBA is not a straight forward. However, the SONIC reduces this task to a simple tallying procedure with its Receive Sequence Numbers (RXpkt.seq\_no). When the Driver detects the LPKT (last packet) bit set to a 1, the sequence numbers indicate how many packets are in a given RBA. Thus, the Driver simply tallies the number of packets processed for a given RBA and when this is equal to the total number of packets, the RBA is free. The sequence numbers are shown below.

15	8	7	0
RBA Sequence Number (Modulo 256)	Packet Sequence Number (Modulo 256)		

If LPKT = 1  
 packet sequence number equals total number of packets minus one in the RBA (packet sequence number starts at zero)

The following three figures (*Figures 1-4a, 1-4b, and 1-4c*) show a scenario depicting the Driver using 3 RBAs and updating the RBA "scoreboard". The flowchart in section 4.2 (*Figure 4-3*) illustrates the recycling of RBAs during receive processing.



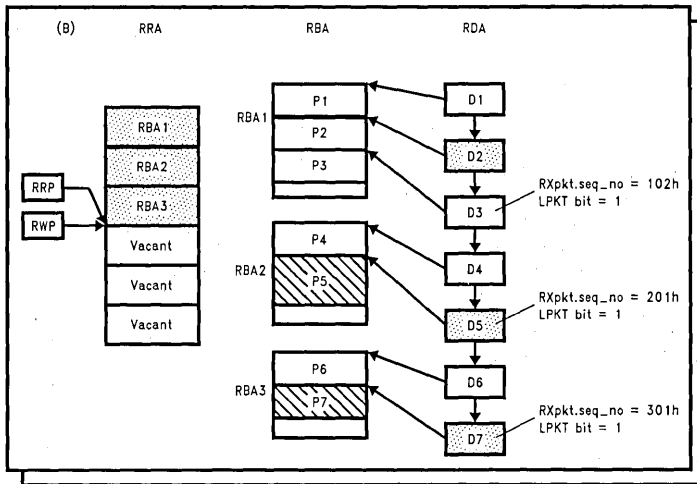
RBA Scoreboard

RBA #	Processed Packets	Total Packets
1	0	3
2	0	2
3	0	2

FIGURE 1-4 (a). Recycling Buffers in the RRA

(a) This figure shows the SONIC, having stored seven packets (P1-P7) in the RBA, has exhausted all its receive buffers (RRP = RWP). The RBA scoreboard indicates that there are 3 unprocessed packets in RBA #1, 2 in RBA #2 and 2 in RBA #3. These numbers are determined by the RXpkt.seq\_no field.

TL/F/11140-6



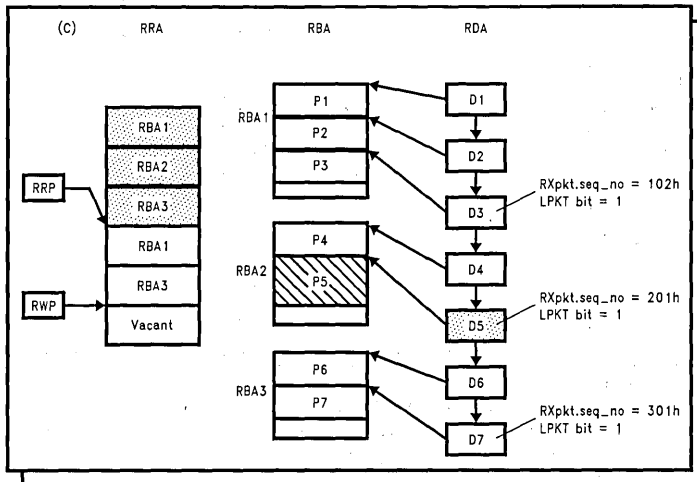
TL/F/11140-7

**RBA Scoreboard**

RBA #	Processed Packets	Total Packets
1	2	3
2	1	2
3	1	2

**FIGURE 1-4(b). Recycling Buffers in the RRA**

(b) The upper level software has finished processing four packets (P1, P3, P4, and P6) and has notified the Driver of this action. The RBA scoreboard now indicates that there is 1 unprocessed packet in RBA #1, 1 in RBA #2 and 1 in RBA #3.



TL/F/11140-8

**RBA Scoreboard**

RBA #	Processed Packets	Total Packets
1	3	3
2	1	2
3	2	2

→ RBA 1 may be recycled

→ RBA 3 may be recycled

**FIGURE 1-4 (c). Recycling Buffers in the RRA**

(c) The upper level has now finished processing 6 packets (P1, P2, P3, P4, P6, and P7). The RBA scoreboard now indicates that RBA #1 and RBA #3 are freed up. The Driver returns these buffers back to the RRA and increments the RWP register accordingly.

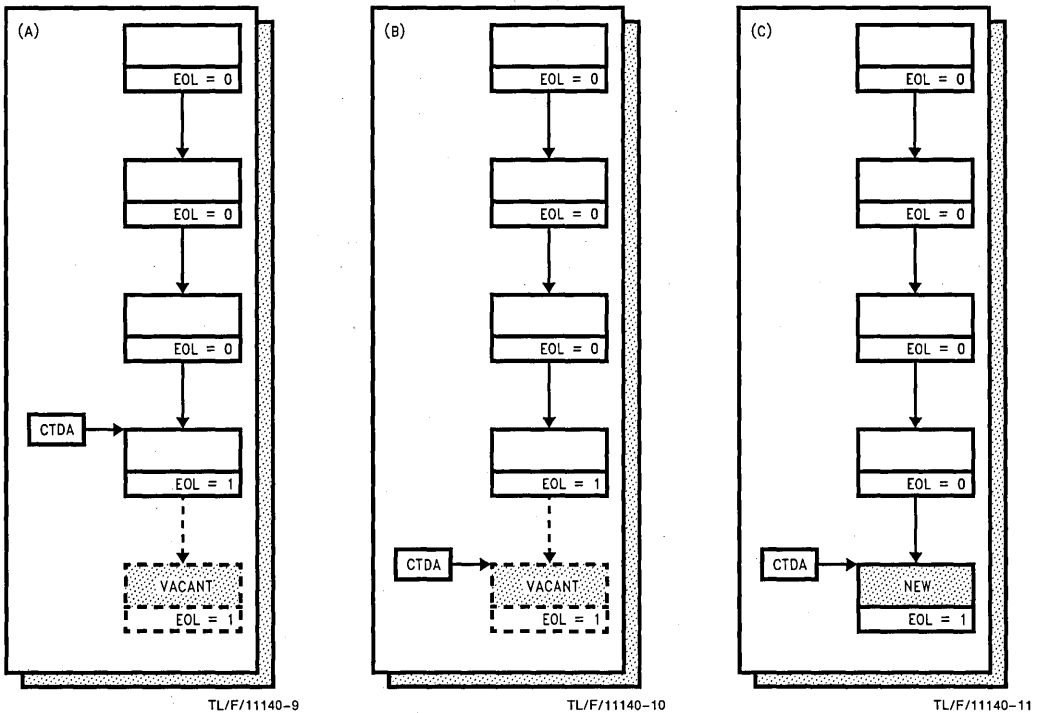
### 1.3 Transmitting Packets from the Transmit Descriptor Area (TDA)

For transmit operation, the Driver uses the TDA to enqueue packets for transmission. Multiple packets may be sent from a single command with each packet allowed to be fragmented (reside in different areas in memory). The fragments themselves may be as small as 1 byte and begin on any byte boundary. Furthermore, particular attention has been made to allow the Driver to append descriptors "on the fly".

To send packets, the driver first creates a list of descriptors in the TDA, then issues the transmit command. The SONIC then reads the TDA and transmits the packets. Once a list is created, the Driver can add to this list "on the fly" without the SONIC stopping. The following rule, however, must be followed: *the last TXpkt.link field must point to the next location where a descriptor will be added* as illustrated in Figure 1-5 (a). The procedure for appending descriptors is outlined as follows:

1. Create a new descriptor with its TXpkt.link pointing to the next vacant descriptor location and its EOL bit set to a "1".
2. Reset the EOL bit to a "0" of the previously last descriptor.
3. Re-issue the Transmit command (setting the TXP bit in the Command register).

Re-issuing the Transmit command assures that the SONIC will continue to send all packets in the list. If the SONIC is currently transmitting, the Transmit command has no effect. If the SONIC has stopped transmitting (which occurs if the SONIC has reached the last descriptor before the Driver has had a chance to append to it) it continues transmitting from where it had previously stopped. The rule, as stated above, guarantees that the Current Transmit Descriptor (CTDA) register points to a valid descriptor after the SONIC has stopped transmitting (see Figures 1-5 (a), (b) and (c)).



**FIGURE 1-5 a, b, c. Appending Descriptors "On the Fly" in the TDA**

- These series of figures shows a scenario whereby the SONIC has reached the end of the descriptor list before the Driver has appended a new descriptor.
- (a) This figure shows the Driver has created a list of four descriptors with the last descriptor pointing to the next location where a descriptor will be added. The transmit command has subsequently been issued and the SONIC has reached the last descriptor.
  - (b) The SONIC has finished transmitting the last descriptor. It reads the last link field and updates the CTDA register to point to the vacant descriptor location. Note that the CTDA register is already prepared for the next transmission.
  - (c) The Driver has appended a descriptor at the vacant location and reissues the transmit command. Note that the CTDA register is pointing to the proper location.

**2.0 REGISTER MODEL OF THE SONIC**

As a brief review, this section gives a short description of the SONIC User registers. This section is similar to section 4.0 of the SONIC datasheet. It may be skipped without loss of continuity.

**2.1 Register Layout**

The SONIC contains 64 16-bit registers used for conveying status and control information. Not all registers, however, are needed by the system since some registers are used for internal operations of the SONIC and others used for in-house factory testing. The registers are categorized as follows:

*User Registers:* The registers are accessed by the user to status, control and monitor SONIC operations. These are the only registers you need to access.

*Internal Use Registers:* These registers are used by the SONIC during the course of operation and are not intended to be accessed by you.

*Factory Test Registers:* These registers are used by National Semiconductor for production testing of the SONIC and should not be accessed. Accessing these registers during SONIC operations may cause erratic behavior.

		RA <5:0>	15	0
Status and Control Registers	0h	Command Register	Status and Control Fields	
	1	Data Configuration Register	Status and Control Fields	
	2	Receive Control Register	Status and Control Fields	
	3	Transmit Control Register	Status and Control Fields	
	4	Interrupt Mask Register	Mask Fields	
Transmit Registers	5	Interrupt Status Register	Status Fields	
	6	Upper Transmit Descriptor Address Register	Upper 16-Bit Address Base	
	7	Current Transmit Address Register	Lower 16-Bit Address Offset	
Receive Registers	2F	Maximum Deferral Timer	Count Value	
	0D	Upper Receive Descriptor Address Register	Upper 16-Bit Address Base	
	0E	Current Receive Address Register	Lower 16-Bit Address Offset	
	14	Upper Receive Resource Address Register	Lower 16-Bit Address Offset	
	15	Resource Start Address Register	Lower 16-Bit Address Offset	
	16	Resource End Address Register	Lower 16-Bit Address Offset	
	17	Resource Read Register	Lower 16-Bit Address Offset	
	18	Resource Write Register	Lower 16-Bit Address Offset	
CAM Registers	2B	Receive Sequence Counter	Count Value	8 7 Count Value
	21	CAM Entry Pointer	4 Pointer	
	22	CAM Address Port 2	Most Signif. 16 Bits of CAM Entry	
	23	CAM Address Port 1	Middle 16 Bits of CAM Entry	
	24	CAM Address Port 0	Least Signif. 16 Bits of CAM Entry	
	25	CAM Enable Register	Mask Fields	
	26	CAM Descriptor Pointer	Lower 16-Bit Address Offset	
Tally Counters	27	CAM Descriptor Count	5 Count Value	
	2C	CRC Error Tally Counter	Count Value	
	2D	Frame Alignment Error Tally	Count Value	
Watchdog Timer	2E	Missed Packet Tally	Count Value	
	29	Watchdog Timer 0	Lower 16-Bit Count Value	
	2A	Watchdog Timer 1	Upper 16-Bit Count Value	
	28	Silicon Revision Register	Chip Revision Number	

**FIGURE 2-1. User Register Grouping**

## 2.2 User Register Grouping

The User register may be further categorized into 6 groups (Figure 2-1) based upon their functionality, i.e., Status and Control, Transmit, Receive, Content Addressable Memory (CAM), Tally counters, and General-Purpose timer. These groups are described as follows:

### 2.2.1 Status and Control Registers

These registers, controlling the transmit, receive, bus, and interrupt operations of the SONIC, consist of the Command, Data Configuration, Receive Control, Transmit Control, Interrupt Mask, and Interrupt Status registers. Of these registers, only the Command and Interrupt Status register are accessed frequently during operation; all others are generally accessed only once during initialization (see section 3.0). These registers are briefly described below.

**Command register.** This register is used for issuing the commands to the SONIC such as transmitting packets, enabling the receiver, and software reset. Commands may be issued by setting the corresponding bit to a "1". During normal operation, the transmit command is the only command that is generally used.

**Data Configuration register:** This register configures the bus interface circuitry, programming the data width size (16 or 32 bits), wait-state insertion (if any), and FIFO threshold. This register may only be written to while the SONIC is in software reset.

**Receive Control register.** This register contains two type of bits, configuration and status. The configuration bits program the SONIC to accept the different classes of packets which may be received such as Physical, Multicast, Broadcast packets, and Runt and Errored packets. The SONIC can also accept all packets from the network for network management and Bridge applications. The Receive Control register also reports the status of the received packet. The software should not read this register directly since status is updated from the next incoming packet and the previous status is overwritten. Instead, the software obtains the status in the status field (RXpkt.status) of the Receive Descriptor Area.

**Transmit Control register.** This register also contains two types of bits, configuration and status. The configuration bits program the various transmit options for (1) generating and interrupts after selected packets have been transmitted, (2) enabling when the "Out of Window" collision timer begins (either at the beginning of the packet or at the State of Frame Delimiter), (3) inhibiting the CRC from being appended to the packet, and (4) enabling the excessive deferral timer (3.2  $\mu$ s). The software should not load this register directly; instead, it writes to the configuration field (TXpkt.config) of the Transmit Descriptor Area (TDA) which the SONIC reads before transmission. The status bits post status of the transmitted packet. Again, this register is not directly read since the SONIC clears the status after it reads the TXpkt.link field. Instead, the software acquires status from the state field (TXpkt.status) in the TDA.

**Interrupt Mask register.** This register enables the various interrupts that the SONIC may generate. Writing a "1" to the bit enables the corresponding interrupt.

**Interrupt Status register.** This register reports interrupts which the SONIC has generated. Interrupts are indicated by a "1" and are cleared when a "1" has been written to it. Since writing a "0" to any bit has no effect, only the specified bits are cleared during the write operation.

### 2.2.2 Transmit Register

The Transmit registers, the Upper Transmit Descriptor Address (UTDA) and the Current Transmit Descriptor Address (CTDA) registers, locate the active descriptor in the Transmit Descriptor Area. The UTDA register, containing a fixed upper 16 bits of address, A<31:16> and CTDA register, containing an active lower 15 bits of address, A<15:1> are concatenated together to form a complete 31-bit address. (The SONIC only provides word or double word addressing.) The LSB of the CTDA register is the End of List (EOL) bit and is used by the SONIC to determine the last descriptor in the list.

### 2.2.3 Receive Registers

The receive registers consist of the Receive Sequence Counter, the End of Buffer Count (EOBC) register, and two groups of registers, the descriptor registers and the resource registers. These registers are briefly described as follows:

**The Receive Sequence Counter:** This counter indicates the number of packets that reside in a particular Receive Buffer Area (RBA). See section 1.1 for an explanation on how to use this register.

**EOBC register.** This register defines the lower boundary in the RBA. If after reception, the remaining numbers words in the RBA are equal to or greater than the EOBC register, reception continues within the same RBA; otherwise, the SONIC stores the packet in another RBA.

**Descriptor registers:** These registers locate the active descriptor in the Receive Descriptor Area (RDA) and are composed of the Upper Receive Descriptor (URDA) and the Current Receive Descriptor (CRDA) registers. These registers are concatenated similarly as the Transmit registers (UTDA and CTDA) above where the URDA contains a fixed upper 16 address bits, A<31:16> and the CRDA contains the lower 15 address bits, A<15:1>. The LSB of the CRDA register is used by the SONIC to determine the last descriptor in the receive list.

**Resource registers:** These registers, used to define the Receive Resource Area (RRA), composed of the Resource Start Area (RSA), the Resource End Area (REA), Resource Write Pointer (RWP), Resource Read Pointer (RRP) and the Upper Receive Resource Address (URRA) registers. The first two registers are static and define the starting and ending points of the RRA. The second two are active and respectively point to the next location where the software places a new descriptor and where the SONIC reads the next descriptor. The SONIC concatenates the last register, the URRA with the other registers to provide a full 31-bit address. The URRA register contains a fixed upper address, A<31:16> and the other four contain an active lower address, A<15:1>. The LSB of these registers is not used since the SONIC only provide word or double word addressability.

**2.2.4 CAM Registers**

The CAM registers are used to access the 16 48-bit CAM entries. Because random accessibility to all CAM entries would consume too much register space (16 x 3 = 48 locations), the CAM entries are accessed via a 4-bit pointer register (CAM Entry Pointer) and 3 16-bit ports (CAM Access Ports 0 to 2). The CAM Entry Pointer selects 1 of 16 entries and the CAM Access Ports 0 to 2, respectively access the least through the most significant portions of the 48-bit entry (Figure 2-2).

**Note:** The least significant byte of the address is the first byte received/transmitted from the network.

**Reading the CAM**

The CAM is accessed in the following manner:

- 1) Place the SONIC in software reset by setting the RST bit in the Command register. This condition must be met before reading the CAM.
- 2) Select the CAM entry by writing the corresponding value in the CAM Entry Pointer.
- 3) Read the CAM Address Ports 0 to 2 to obtain the complete 48-bit entry.

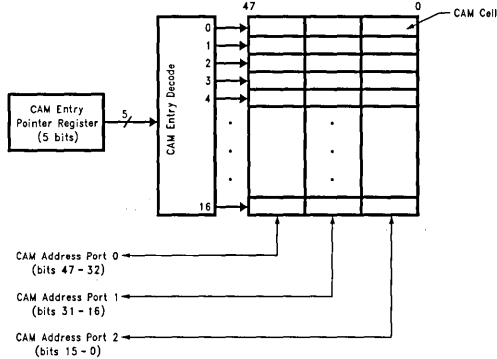
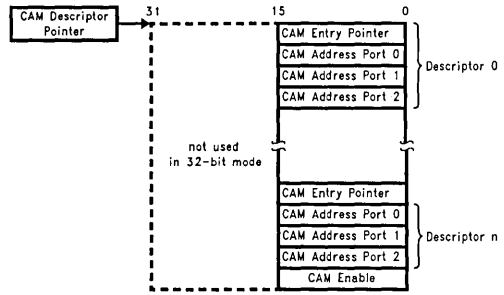


FIGURE 2-2. CAM Organization

**Writing to the CAM**

To avoid internal conflicts with the CAM entries when receiving packets, the SONIC does not allow the entries to be written to directly. Instead, the entries are written to indirectly via the CAM Descriptor Area (CDA). This area, maintained in memory, contains the data to be written into the CAM and upon command, the SONIC reads this area and load its CAM. The CDA is composed of *n* number descriptors (Figure 2-3) which are used to load the CAM Entry Pointer, the CAM Access Ports, and the CAM Enable register. To program the CAM, you first initialize the CDA, load the CAM Descriptor Count register with the number of descriptors and the CAM Descriptor Pointer register with the starting address of the CDA, then issue the Load CAM command to the SONIC. This operation is summarized below:

- 1) Load the CDA as specified in Figure 2-3.
- 2) Load the CAM Descriptor Count register with the number of descriptors.
- 3) Load the CAM Descriptor Pointer register with the starting address of the CDA.
- 4) Issue the Load CAM command (setting the LCAM bit in the Command register). The SONIC finishes this command when the LCAM bit is reset.



TL/F/11140-13

FIGURE 2-3. CAM Descriptor Area Format

**2.2.5 Tally Counters**

The Tally counters maintain the network management events which occur too frequently for the software to maintain. These events, CRC errors, frame alignment errors, and missed packets are tallied by the CRC, FAE and Missed Packets Tally counters. These counters are 16-bit counters and can generate an interrupt when a rollover occurs. These registers are generally used in conjunction with software to maintain a 32-bit counter. These counters maintain the time-sensitive lower 16 bits of the count while software maintains the upper 16 bits.

**2.2.6 General-Purpose Timer**

This 32-bit timer, clocked at one half the 10 MHz transmit clock frequency, is used for timing user definable events. The timer measures events ranging from microseconds up to minutes. The time can be calculated by multiplying the count value by 200 ns (1/2 the transmit clock period). Table 2-1 gives some example values. To use the timer, you first load the timer with a count value, then start the timer by setting the ST bit in the Command register. The SONIC then begins decrementing the timer. When the rollover is reached (0000 0000h to FFFF FFFFh), the Timer Complete (TC) bit in the Interrupt Status register is set. Note that the timer does not stop when the rollover occurs, but continues to decrement (from FFFF FFFFh). It must be explicitly stopped by setting the STP bit in the Command register.

Table 2-1. Example Timer Values

Timer	WT1	WT0
0.1 sec	7	A120
0.5 sec	26	25A0
1.0 sec	4C	4B4
10 sec	2FA	F080
30 sec	8F0	D180
1 min	11E1	A300
5 min	5968	2F00
10 min	B2D0	2E00

**2.2.7 Silicon Revision Register**

This register supplies information on the revision stepping of the SONIC. This register begins at zero and counts upward. Contact National Semiconductor for latest information on this register.

### 3.0 INITIALIZING THE SONIC

Initializing the SONIC is the crucial first step before any SONIC operations can commence. This step involves setting up the SONIC's registers for reception and transmission and initializing the memory structures for the Buffer Management. This section describes the initialization process by introducing what information is needed, then discussing an example initialization routine.

#### Getting Started

Before initializing the SONIC, a few details regarding the hardware and network operating system must be obtained. By answering the questions below, the required information can be gathered.

1) What is the bus size?

The SONIC supports bus sizes of 16 or 32 bits.

2) Does the system operate in a synchronous or asynchronous manner?

This question refers to how the  $\overline{\text{RDY}}$  (or  $\overline{\text{DASCKO}}$ , 1) input is issued to the SONIC. If this line is asserted with guaranteed setup and hold times by the hardware, use synchronous mode; otherwise, use asynchronous mode. Synchronous mode has the advantage of having a minimum memory cycle of 2 bus clocks as opposed to 3 bus clocks for asynchronous mode.

3) What is the maximum bus latency does the SONIC expect?

The bus latency is the time from when the SONIC requests for the bus (by asserting the HOLD or  $\overline{\text{BR}}$  pin) to when the SONIC begins using the bus. The bus latency tolerance can be increased by programming the transmit FIFO threshold higher and the receive FIFO lower. The bus latency tolerance is calculated by the following equations:

$$\text{TX FIFO Tolerance} = (\text{FIFO threshold}) \\ * (0.8 \mu\text{s})$$

$$\text{RXFIFO Tolerance} = (32 - \text{FIFO threshold}) \\ * (0.8 \mu\text{s})$$

4) Do wait-states need to be added into the memory cycle?

The SONIC can operate up to a 2 bus clock memory cycle. If this is too fast, you can program the SONIC to insert 1 to 3 wait-states for each memory cycle. A two clock memory cycle requires a memory access time of approximately 40 ns–50 ns. (Note that wait state can also be inserted by hardware using the  $\overline{\text{RDY}}$  or  $\overline{\text{DSACKO}}$ ,  $\overline{\text{T}}$  inputs.)

5) What type of packets do you want to accept?

The SONIC is generally programmed to accept its own physical address and the Broadcast address. In some applications, however, the SONIC may be programmed to accept multiple physical/multicast addresses (up to 16), and errored and runt packets.

6) What is the maximum number of consecutive packets that you expect to receive?

This question is perhaps the most difficult to answer since it deals with the upper level protocols. In many transport protocols, flow control is used by the receiving node to limit the number of consecutive packets the transmitting node may send unacknowledged. This is generally called the "window size". Ideally, the software provides the SONIC with the memory resources it needs to completely buffer a complete "window".

7) What types of interrupts do you want the system to respond to?

The SONIC can generate a variety of interrupts. Not all interrupts, however, need be (or should be) used to generate interrupts to the system. For maximum performance, you want as few interrupts as possible. A typical system allows interrupts occurring from good receptions and transmissions, and errored transmissions.

#### Initialization Example

Once the above questions have been answered, you can begin coding the initialization routine. This routine has been divided into 9 steps, but, only steps 1 and 9 need to be followed in the order presented. Example code is provided in the appendix.

1) Reset the SONIC: When the SONIC is powered-on, the hardware generally resets the SONIC by pulsing the RESET pin low. Thus, software does nothing to reset the SONIC. Once reset, the SONIC remains in reset mode until the RST bit in the Command register is cleared. If the hardware does not provide the reset, the software can perform the functional equivalent by simply setting the RST bit. All initialization should be done in reset mode to prevent spurious actions by the SONIC.

2) Configure the System Interface: This step writes to the Data Configuration Register (DCR) to configure the SONIC's bus interface circuitry. The configuration information is found by answering questions 1 through 4, discussed above. Note that the DCR can only be written to in reset mode.

3) Set Up the Receive Filters: This step determines what types of packets to accept (i.e., Physical, Multicast, Broadcast, Runt, and Errored packets) and what addresses to accept. The type of packet to accept is programmed in the Receive Configuration register and the addresses to accept are programmed into the Content Address Memory (CAM). See section 2.2.4 for loading the CAM.

4) Enable the Interrupts: This step enables the interrupts by writing to the Interrupt Mask register (IMR). Note that the interrupting condition is indicated by the Interrupt Status Register (ISR), but will not generate an interrupt unless the corresponding IMR bit is set. Note also that if the SONIC is initialized in reset mode, no interrupts can be generated.

5) Initialize Memory: This step initializes the three memory structures used by the SONIC for transmission and reception and allocates the memory blocks for storing received packets. An initialization example is illustrated in Figures 3-1 and 3-2. The *non-shaded* areas indicate fields which must be initialized and *shaded* areas indicate fields which are written to by the SONIC.



There are a few caveats discussed below:

All Descriptor Areas:

- Descriptor must be aligned to word (16-bit) boundaries in 16-bit mode and aligned to double word (32-bit) boundaries in 32-bit mode.
- The Descriptor Areas must not cross over a 32k word boundary since it only operates within this range.
- In 32-bit mode, the upper 16 data bits, D<31:16> are not used.

Transmit Descriptor Area:

- The transmit buffers (Transmit Buffer Area) may be aligned to any boundary; that is, the TXpkt.ptr0, 1 fields may contain any value.
- The packet and fragment size may be as low as 1 byte; that is, the TXpkt.pkt\_size and TXpkt.frag\_size may contain the value of one.

Receive Resource Area

- The resource descriptors must be contiguous and can not straddle the endpoints.
- In the lower buffer pointer field, RXsrc.ptr0, the SONIC ignores least significant bit in 16-bit mode and the 2 least significant bits in 32-bit mode. This forces receive buffers to always align to either word or double word boundaries.

6) Initialize the Buffer Management Registers: This step initializes the buffer management registers to the starting positions in the buffer management (see *Figures 3-1* and *3-2*). These initialized registers are shown in Table 3-1.

7) Issue RRA command: By setting the RRRRA bit in the Command register, you force the SONIC to read the RRA. The SONIC reads the RRA beginning at the RRP location and loads the following registers. (For mnemonics description, see appendix.)

CRBA0 ← RXsrc.ptr0

CRBA1 ← RXsrc.ptr1

RBWC0 ← RXsrc.wc0

RBWC1 ← RXsrc.wc1

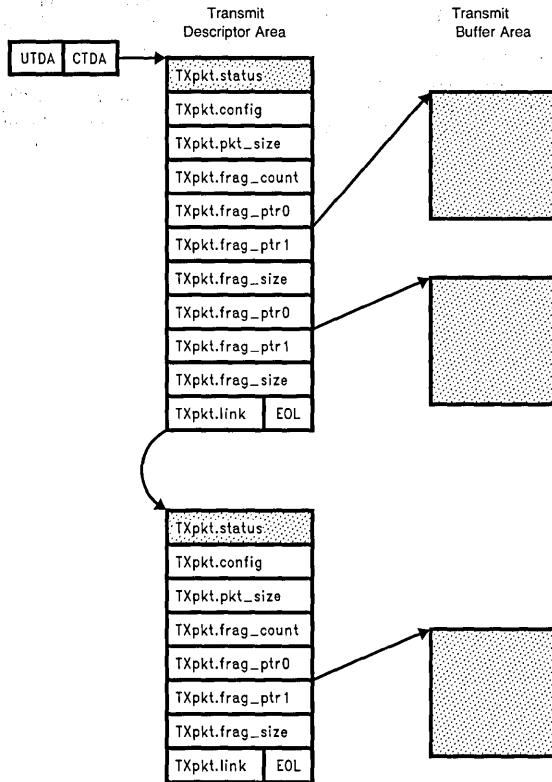
After this command has executed (RRRA bit resets), the SONIC is ready to store the next packet in the first RBA allocated to it.

8) Clear and Tally Counters (optional): The tally counters (CRC, Frame Alignment, and Missed Packets) may be cleared by writing FFFFh to these registers. These counters will rollover after FFFFh is reached.

9) Bring the SONIC On-line: This last step commissions the SONIC to receive, transmit, and generate interrupts. The software enables the SONIC by setting the RXEN bit and clearing the RST bit in the Command register.

**TABLE 3-1. Initialization of Buffer Management Registers**

Reg.	Initialized with
URDA	A <31:16> of starting location of RDA
CRDA	A <15:1> of starting location of RDA
UTDA	A <31:16> of starting location of TDA
CTDA	A <15:1> of starting location of TDA
URRA	A <31:16> of starting location of RRA
RSA	A <15:1> of starting location of RRA
REA	A <15:1> of ending location of RRA
RRP	Points to first descriptor the SONIC reads
RWP	Points to next location where the software will place a descriptor



**FIGURE 3-1. Initialization Example for Transmit Buffer Management (shaded areas not initialized)**

TL/F/11140-14

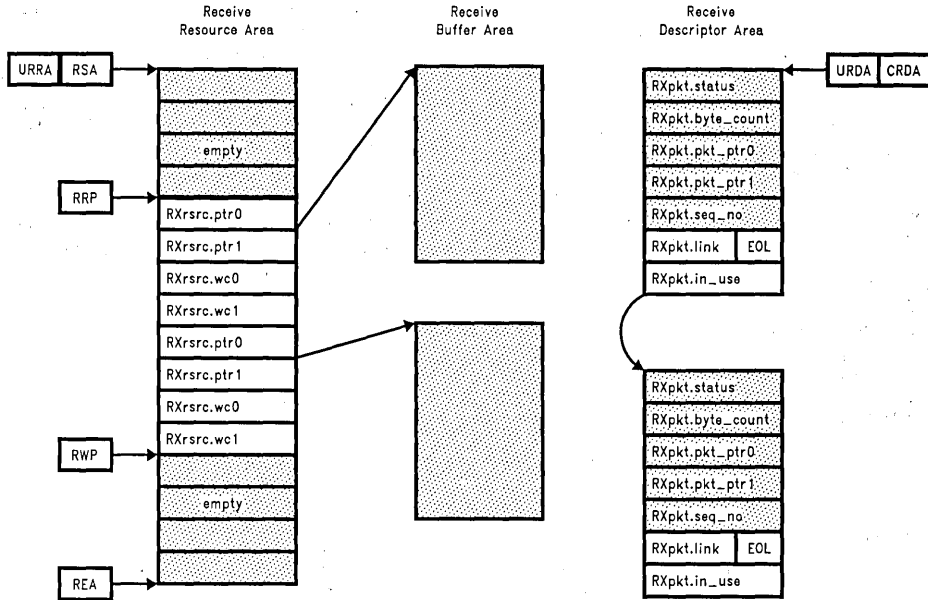
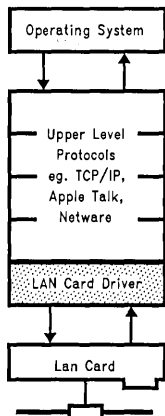


FIGURE 3-2. Initialization Example for Receive Buffer Management (shaded areas not initialized)

TL/F/11140-15

4.0 WRITING DRIVERS FOR THE SONIC

The Driver (see Figure 4-1), being the lowest level of software, shields the upper software levels from the details of the hardware. The Driver performs the required low-level transmit and receive functions such as passing packet up to the upper level software, recycling receive buffers, and enqueueing packets for transmission. The Driver performance is important since it may potentially receive packets at the full network rate. Any packet losses at this level can severely affect the overall performance of the network. This section describes the basic algorithms for writing a Driver for the SONIC. Example code is provided in the appendix.



TL/F/11140-16

FIGURE 4-1. Relationship of Driver of Upper Level Software

Overview

The Driver for the SONIC consist of two procedures, INITIATE\_TX (Figure 4-2) and SONIC\_ISR (Figure 4-3) for transmit and receive operations. During transmit operations, the upper level software first assembles packets for transmission by gathering the pointers to the fragments and then calling INITIATE\_TX to begin the transmission. When the SONIC finishes transmission, it interrupts the system. The system then enters the interrupt service routine, SONIC\_ISR, where it reports the status of the packets transmitted. During received operations, the SONIC also interrupts the system upon receiving a packet. The system enters SONIC\_ISR to post status and then to pass the packet up to the upper level software via pointers.

4.1 INITIATE\_TX

This procedure requires that all pointers to the fragments and the sizes of these fragments are passed down to it by the upper level software. It only initiates a packet for transmission; it does not report status. This action is performed by SONIC\_ISR after the packet has been transmitted. INITIATE\_TX operates as follows:

- 1) Obtains the pointers delivered by the upper level software and fills out a descriptor in the Transmit Descriptor area (TDA).
- 2) If the packet is less than 64 bytes, it pads it out to this length.
- 3) Issue the transmit command to the SONIC and return.

It is important that descriptors are appended in the manner prescribed in section 1.3. This algorithm improves performance by guaranteeing that the SONIC continues to transmit all packets in the descriptor list.

1

## 4.2 SONIC\_ISR

This procedure is the interrupt service routine which responds to three interrupts generated by the SONIC: PACKET RECEIVED, TRANSMISSION DONE, and TRANSMIT ERROR. Interrupts occurring before and during the interrupt service routine are serviced before SONIC\_ISR exits. SONIC\_ISR is broken down into three main sections: (1) reading the cause of the interrupt, (2) processing received packets, and (3) posting status of transmitted packets. The first action performed is finding the cause of the interrupt. For receive interrupts, SONIC\_ISR jumps to the receive routine, and for transmit interrupts (good and errored transmissions), it jumps to the transmit routine. The receive routine examines the first descriptor in the RDA, then passes the pointer of the packet up to the upper level software for further processing. It continues reading the RDA until it reaches the end of the descriptor list: The receive routine also recycles receive buffers as necessary. The transmit routine reads the first descriptor in the TDA and reports the status of the transmitted packet to the upper level software. If more than one packet has been enqueued, the transmit routine examines the complete list in the TDA. SONIC\_ISR is summarized below.

### Reading the Interrupt

- 1) Read the Interrupt Status register for the cause of interrupt. If a transmit interrupt has occurred, go to step 2; if a receive interrupt has occurred, go to step 4; or if no more interrupts are present, return.

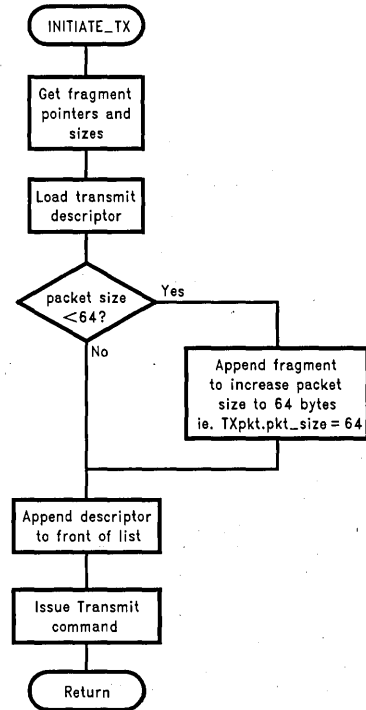
### Transmit Routine

- 2) Read the next TXpkt.status in the Transmit Descriptor Area and post status to the upper level software.
- 3) Read the End of List (EOL) bit in the TXpkt.link field to determine if the current descriptor is the last descriptor. If it is not, go back to step 2 to post status of the remaining packets; otherwise go back to step 1.

### Receive Routine

- 4) Read the next RXpkt.status field in the Receive Descriptor Area and pass the pointer and status of the packet up to the upper level software.
- 5) Read the RXpkt.seq\_no field. If the RBA number is different from the previous one, enter the RBA number into the RBA "scoreboard". For more information, see section 1.2.
- 6) Check the LPKT bit from the RXpkt.status field. If set to "1", enter the packet sequence number (from the RXpkt.seq\_no) into the RBA scoreboard.

- 7) Read the RXpkt.in\_use field, if the field is cleared to all zeros, go back to step 4 to process the remaining packets; otherwise if RXpkt.in\_use is not equal to zero, the end of the list has been reached; proceed to step 7.
- 8) Call the system to determine which packets have been processed by the upper level software. Tally the processed packets in the RBA scoreboard.
- 9) Find freed up RBAs and return them to the front of Receive Resource Area (RRA).
- 10) Find the freed up receive descriptors and return them to the front of the descriptor list; then go to step 1.



TL/F/11140-17

FIGURE 4-2. INITIATE\_TX Routine

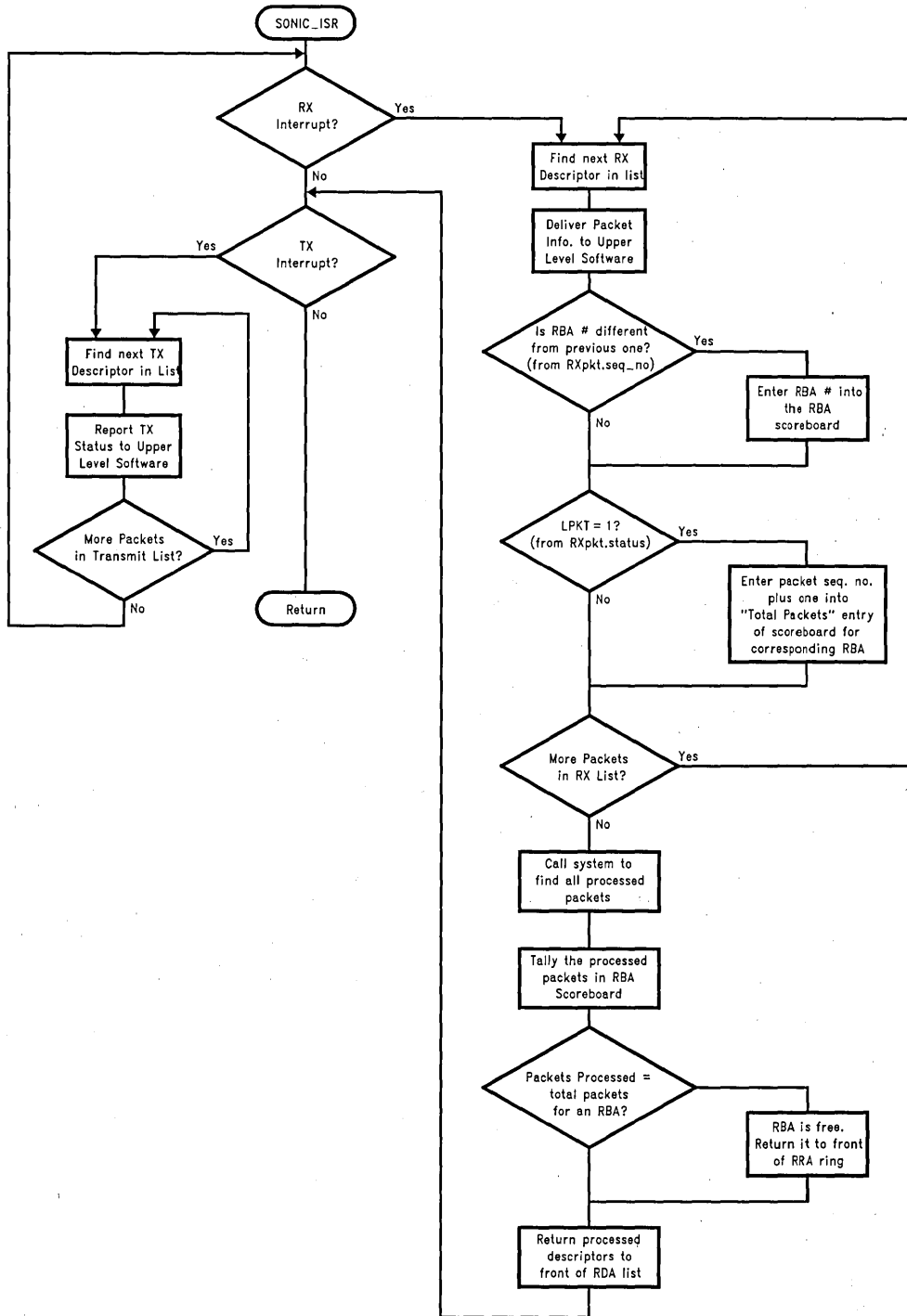


FIGURE 4-3. SONIC\_ISR Routine

TL/F/11140-18

## 5.0 STRATEGIES FOR IMPROVING DRIVER PERFORMANCE

Making the Driver as efficient as possible is crucial for the overall performance of the network. Empirical results have shown that the difference between a poor and a good Driver can vary as much as 10% to 20%. The Driver is particularly vulnerable to becoming a bottleneck since it may, at times, be receiving data at the full network bandwidth (10 Mb/s). Any packets that are lost at the Driver level impacts all levels. While upper level protocols provide packet recovery mechanisms, these tend to be quite slow (on the order of seconds). Typically, software timers must time out before the upper level software retransmits an unacknowledged packet. In this section, some hints are discussed to make a fast Driver.

- 1) Write the Driver in assembly code: The fastest code is generally written in assembly code since people write more efficient code than a compiler. Writing your own assembly code also gives you the option to use some "tricks" which are not normally accepted as "good" programming practice. One such example is using a JUMP statement instead of a CALL statement. The JUMP statement, by nature, is quite messy, but is considerably faster since it involves less CPU cycles. Of course, the disadvantage in using assembly code is that it is less readable and portable. As a compromise, you may consider a good optimizing compiler.
- 2) Reduce the Number of Interrupts: Interrupts to the system inherently make it less efficient since the CPU must make a context switch between what it was currently doing to the interrupt service routine. This switch involves pushing the CPU registers onto the stack, jumping to an interrupt vector table, issuing an interrupt acknowledge to the interrupt controller, then executing the interrupt service routine. The overhead associated with each interrupt makes the CPU less efficient. The example interrupt service routine discussed in section 4.0, responded to interrupts generated from good transmission and receptions, and errored transmission. It is possible, however, to reduce the source of interrupts to just two, allowing only interrupts to occur from good receptions and errored transmissions. The reason good transmission interrupts may be eliminated is because the upper level software generally does nothing for these events. Only for an errored transmission must the upper level software intervene such as to retransmit the packet. Good transmissions, while they still need to be reported, can be status on a less timely basis such as after processing receive interrupts or after a specified time period. The SONIC's General Purpose timer can be used to generate such a time period.
- 3) Append Transmit Descriptors as described in section 1.0: The algorithm described guarantees that the SONIC continues to transmit all packets in the list, even if it has reached the point where the new descriptor(s) have been appended to the end of the list. If the algorithm is not followed, the SONIC may stop at the enjoining point and this forces the Driver to intervene.
- 4) Supply Sufficient Number of Receive Packet Descriptors: Since the receive descriptor uses a relatively small amount of memory (7 words or double words, depending on the data size mode), allocate sufficient number of them such that the SONIC never (or at least rarely) runs

out of them. If the SONIC ever runs out of them, reception ceases, resulting in packet losses. The number of descriptors to allocate can be determined by answering question 6 of section 3.0.

- 5) Make the Receive Resource Area (RRA) Sufficiently Large: Since the RRA does not take up much memory (4 words or double words per descriptor), make it larger than the total number descriptors you expect to put into it. For example, if you expect you will need 10 resource descriptors, make the RRA large enough to accommodate 15 descriptors. Making the RRA larger than you will need, prevents the RRA from becoming a bottleneck in adding more resources.
- 6) Optimize the Size of the Receive Buffer Areas (RBAs): Generally speaking, the larger the RBAs, the more efficient the Driver. This is because the Driver handles fewer number of receive buffers and, thus, less processing time is dedicated to managing the buffers. There is a tradeoff, however. If the buffers are very large, the entire buffer areas are locked out for recycling so that large buffers become less space efficient in memory. As a guideline, 4k to 8k byte RBAs are good starting points for experimentation. Use larger buffers, if memory is plentiful.

## 6.0 SELF-TEST DIAGNOSTICS

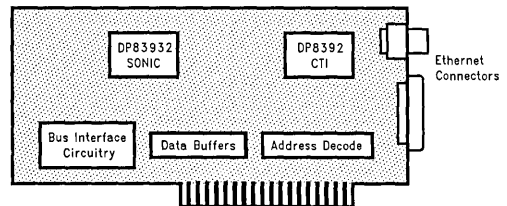
After the hardware has been designed and the Drivers written, there is still a need to verify that the hardware is still functioning. Rough shipping or improper handling (without static protection) can produce innumerable problems. Some boards which work fine in the lab invariably fail in the field. Thus, self-test diagnostics are used to determine the health of the boards and diagnose problems if something is amiss.

Figure 6-1 shows the basic components of the Ethernet system: address decode circuitry, data buffers, bus interface logic, Ethernet chipset (SONIC and transceiver) and the Ethernet connectors (BNC and 15-pin D). The Ethernet hardware can be fully tested by using the SONIC's three loopback modes. Each loopback mode is full-duplex, transmitting data as well as receiving it and are summarized below. An example routine is given in the appendix.

Mode 1: Data is routed back through the SONIC's MAC Unit. Both the transmit and receive Buffer Management operations are active and must be initialized accordingly. Verifies the MAC Unit, Bus interface logic, address decode circuitry and data buffers.

Mode 2: Similar to above, but data is routed back through the SONIC's ENDEC Unit. Verifies the SONIC's ENDEC unit.

Mode 3: Similar to above, but data is routed back at the transceiver. Verifies the Ethernet connectors (BNC and 15-pin D) and Ethernet transceiver (DP8392 CTI).



TL/F/11140-19

FIGURE 6-1. Basic Components of Ethernet Hardware

## Appendix

## A. Initialization Routine

```

/*****
/*
/* Initialization Routine for SONIC
/*
/*****

sonic_init ()
{
    unsigned short init_RRA[512]; /* memory for RRA */

    /* initialize some registers */
    set_reg_value();

    /* allocate memory for TX descriptors and init UTDA and CTDA */
    init_tda()

    /* Init receive buffer area and RX registers */
    Init_Des_page();
    Initial_RRA(RRA_NUM);
    Init_RDA(RDA_NUM);

    /* Issue Read RRA command */
    /* Must first bring SONIC out of reset before issuing any
       commands */
    REG_WRITE(card.crd_iobase+SONIC_cr*2, 0x0);
    REG_WRITE(card.crd_iobase+SONIC_cr*2, 0x0100);

    /* Bring SONIC on-line by enabling MAC receiver */
    REG_WRITE(card.crd_iobase+SONIC_cr*2, 0x0008);
}
/*****
/* This routine initializes some of the SONIC's registers.
/* ie., CR, DCR, RCR, IMR, ISR, CRCT, FAET, and MPT
/*
/*****
set_reg_value()
{

    /* Put SONIC is reset */
    REG_WRITE(card.crd_iobase+SONIC_cr*2, 0x0080);

/* dcr value depends upon data width (16 or 32 bits) */
#ifdef BIT32
    REG_WRITE(card.crd_iobase + SONIC_dcr*2, 0x00f9);
#else
    REG_WRITE(card.crd_iobase + SONIC_dcr*2, 0x00d9);
#endif

    REG_WRITE(card.crd_iobase + SONIC_rcr*2, 0x0000);
    REG_WRITE(card.crd_iobase + SONIC_imr*2, 0x3fff);
/* Clear ISR */
    REG_WRITE(card.crd_iobase + SONIC_isr*2, 0xffff);
/* Clear Tally counters by writing FFFFh to them */
    REG_WRITE(card.crd_iobase + SONIC_crct*2, 0xffff);
    REG_WRITE(card.crd_iobase + SONIC_faet*2, 0xffff);
    REG_WRITE(card.crd_iobase + SONIC_mpt*2, 0xffff);
}

```

TL/F/11140-20

```

/*****
*
* Allocate memory for TDA and initialize UTDA and CTDA registers.
*
*****/

init_tda()
{
    short i;
    unsigned long addr;
    unsigned long tda1_start, tda2_start, tda3_start;
    unsigned short saddr;
    unsigned short ul6, l16;

    /* Allocate memory for TDAs */
    tda1=(ONE_FRAG_TDA *) malloc(sizeof(ONE_FRAG_TDA) + 2);
    tda1_start = (unsigned long) tda1;
    tda2=(TWO_FRAG_TDA *) malloc(sizeof(TWO_FRAG_TDA) + 2);
    tda2_start = (unsigned long) tda2;
    tda3=(TWO_FRAG_TDA *) malloc(sizeof(TWO_FRAG_TDA) + 2);
    tda3_start = (unsigned long) tda3;

    /* Force TX descriptors to double word alignment */
#ifdef BIT32
    if ( (tda1_start & 0x00000003) == 0)
        ;
    else
        tda1_start += 2;
    if ( (tda2_start & 0x00000003) == 0)
        ;
    else
        tda2_start += 2;
    if ( (tda3_start & 0x00000003) == 0)
        ;
    else
        tda3_start += 2;
#endif
    /* Convert the double word alignment address to pointer */
    tda1=(ONE_FRAG_TDA *) tda1_start;
    tda2=(TWO_FRAG_TDA *) tda2_start;
    tda3=(TWO_FRAG_TDA *) tda3_start;

    /* Finding effective address of TDA1 to load UTDA and CTDA regs.*/
    addr=(unsigned long) tda1; /* Using large mem. model.*/
    /* addr is the address in 8086 format */
    /* upper 16 bits = BASE, lower 16 bits = OFFSET */
    ul6 = addr >> 16;
    l16 = addr;
    addr=(unsigned long) ul6 * 16 + l16;
    ul6 =addr >> 16;
    REG_WRITE(card.crd_iobase+SONIC_utda*2, ul6);
    REG_WRITE(card.crd_iobase+SONIC_ctda*2, addr);
}

```



```

/*****
Name
    Initialize Descriptor Page
Syntax
    Init_Des_Page();
Description
    This function gets 3 4K consecutive bytes of memory
    from the host for the RBA. Also initializes the URRR
    and CDP registers.

Input
    None.

Author
    Michael Lui

```

```

*****/

short Init_Des_Page()
{
    unsigned short urra;    /* upper 16 bits of the beginning
                           addr of RRA */
    unsigned short urda;    /* upper 16 bits of the beginning
                           addr of RDA */
    unsigned short cdp;     /* beginning address of the cdp */
    R_DESCRIPTOR *temp_RDA;
    unsigned short i;       /* index */
    unsigned long laddr;
    unsigned long addr;
    long EA();
    unsigned long rba1_start, rba2_start, rba3_start;
    unsigned short ul6, l16;

    /* allocate memory to RDAs */
    L_RDA=F_RDA=NULL;
    for (i=0; i<RDA_NUM; i++) {
        temp_RDA=(R_DESCRIPTOR *) malloc(sizeof(R_DESCRIPTOR) + 2);

        /* force double word alignment for RX descriptor */
#ifdef BIT32
        addr = (unsigned long) temp_RDA;
        if ((addr & 0x00000003) == 0)
            ;
        else
            addr += 2;
        temp_RDA = (R_DESCRIPTOR *) addr;
#endif
        temp_RDA->next=NULL;
        if (F_RDA == NULL)
            L_RDA=F_RDA=temp_RDA;
        else {
            L_RDA->next=temp_RDA;
            L_RDA=temp_RDA;
        }
    }
    /* allocate memory for RBA */
    init_RBA1=(unsigned char *) malloc(4100);

```

TL/F/11140-22

```

init_RBA2=(unsigned char *) malloc(4100);
init_RBA3=(unsigned char *) malloc(4100);

rba1_start=(unsigned long) init_RBA1;
rba2_start=(unsigned long) init_RBA2;
rba3_start=(unsigned long) init_RBA3;

/* forcing double word alignment for RBAs. */
#ifdef BIT32
if ( (rba1_start & 0x00000003) == 0)
;
else
rba1_start+=2;
if ( (rba2_start & 0x00000003) == 0)
;
else
rba2_start+=2;
if ( (rba3_start & 0x00000003) == 0)
;
else
rba3_start+=2;
#endif

/* Convert double word alignment address to pointer */
RBA1 = (unsigned char*) rba1_start;
RBA2 = (unsigned char*) rba2_start;
RBA3 = (unsigned char*) rba3_start;

/* initialize URRA and CDP registers */
RRA_start = (unsigned long) init_RRA;
/* check RRA is aligned on double word boundary */
#ifdef BIT32
if ( (RRA_start & 0x00000003) == 0)
;
else
RRA_start+=2;
#endif

/* Assign urra */
laddr = (unsigned long) RRA_start;
u16=laddr >> 16;
l16=laddr;
laddr = (unsigned long) u16 * 16 + l16;
urra = laddr >> 16;
/* Load the URRA register */
REG_WRITE(card.crd_iobase+SONIC_urra*2, urra);

/* load the CDA descriptor pointer */
laddr = (unsigned long)u16 * 16 +l16 +CAM_OFFSET;
cdp=laddr;
/* load the CDP register */
REG_WRITE(card.crd_iobase+SONIC_cdp*2,cdp);
}

```

```

/*****
Name
Initialize RRA

Syntax
flag=Init_RRA(n);

Description
This function will create a circular queue with n
number of RRA descriptors in it. The RRA descriptors
are pointing to the corresponding RBA blocks. It will
also load the RSA, REA, RRP, and RWP registers.

Returned Value
1 = Success
0 = Failed

Author
Michael Lui

```

```

*****/

```

```

short Initial_RRA()
{
    struct sonicreg *sonic=0;
    unsigned short rsa;          /* Resource Start Area */
    unsigned short rea;          /* Resource End Area */
    unsigned short rrp;          /* Resource Read Pointer */
    unsigned short rwp;          /* Resource Write Pointer */
    unsigned short urba;        /* Upper 16 bit of the RBA starting
                                address */
    unsigned short lrba;        /* Lower 16 bit of the RBA starting
                                address */
    unsigned short i;           /* for loop index */
    unsigned short low_addr;
    unsigned short high_addr;
    unsigned long addr,laddr;
    short inc;                  /* RRA increment */
    unsigned short ul6, ll6;

    addr = (unsigned long) RRA_start;
    ul6=addr >> 16;
    ll6=addr;
    addr = (unsigned long) ul6 * 16 + ll6;

    /* Lower 16 bit of the RRA */
    rsa = (unsigned short) addr;
    /* Load the RSA Register */
    REG_WRITE(card.crd_iobase+SONIC_rsa*2, rsa);

    laddr=addr + RWP_OFFSET;
    rea = (unsigned short) laddr; /* Ending address of RRA */
    /* Load the REA Register */
    REG_WRITE(card.crd_iobase+SONIC_rea*2, rea);

    rrp = rsa; /* Read Pointer starts at the beginning
                address */

    /* Load the RRP Register */

```

```

REG_WRITE(card.crd_iobase+SONIC_rrp*2, rrp);

laddr = addr + RWP_OFFSET/2;
rwp = (unsigned short) laddr; /* Only 3 descriptors
                               initially */
/* Load the RWP Register */
REG_WRITE(card.crd_iobase+SONIC_rwp*2, rwp);

/* Initialize the RRA descriptors */
RRA=RRA_start;
/* for 32-bit memory each descriptor uses a double word, for
16-bit memory, each descr. uses a word. */
#ifdef BIT32
    inc=4;
#else
    inc=2;
#endif
/* Load RBA1 address */
addr=(unsigned long) RBA1;
u16=addr >> 16;
l16=addr;
addr=(unsigned long)u16 * 16 + l16;
low_addr = addr & 0x0000ffff;
* (unsigned long *)RRA = low_addr;
RRA +=inc;
* (unsigned long *)RRA = addr >> 16;
RRA +=inc;
/* Load RXrsrc.buff_wc0 */
* (unsigned short *)RRA = 0x0800;
RRA +=inc;
/* Load RXrsrc.buff_wcl */
* (unsigned short *)RRA = 0;
RRA +=inc;

/* Load RBA2 address */
addr=(unsigned long) RBA2;
u16=addr >> 16;
l16=addr;
addr=(unsigned long) u16 * 16 + l16;
low_addr = addr & 0x0000ffff;
* (unsigned short *)RRA = low_addr;
RRA +=inc;
* (unsigned short *)RRA = addr >> 16;
RRA +=inc;
/* Load RXrsrc.buff_wc0 */
* (unsigned short *)RRA = 0x0800;
RRA+=inc;
/* Load RXrsrc.buff_wcl */
* (unsigned short *)RRA = 0;
RRA +=inc;

/* Load RBA3 address */
addr=(unsigned long) RBA3;
u16=addr >> 16;
l16=addr;
addr=(unsigned long)u16 * 16 + l16;
low_addr = addr & 0x0000ffff;
*(unsigned short *)RRA = low_addr;
RRA+=inc;
* (unsigned short *)RRA = addr >> 16;

```

```
RRA+=inc;
/* Load RXrsrc.buf_wc0 */
* (unsigned short *)RRA = 0x0800;
RRA+=inc;
/* Load RXrsrc.buf_wc1 */
* (unsigned short *)RRA = 0;
RRA+=inc;
}
```

TL/F/11140-26

```

/*****
Name
Initialize RDA

Syntax
flag = Init_RDA(n);

Description
This function will create a linked list of some
arbitrary number of packet descriptors. The EOL bit for
the last descriptor should set to 1 while the others
should set to 0. The in_use field should set to a
non-zero value for all descriptors. The CRDA register
should loaded with the address of the first descriptor.

Returned Value
1 = Success
0 = Failed

*****/
short Init_RDA()
{
unsigned long crda; /* Current CRDA Register */
unsigned char *RDA; /* RDA address */
R_DESCRIPTOR *cur_RDA; /* current RDA */
unsigned short n_RDA_addr; /* next RDA address */
unsigned long addr;
short i;
unsigned u16, l16;

crda = (unsigned long) F_RDA;
u16 = crda >> 16;
l16 = crda;
crda = (unsigned long)u16 * 16 + l16;

/* Load the CRDA Register */
REG_WRITE(card.crd_iobase+SONIC_crda*2, crda);

cur_RDA=F_RDA;
while (cur_RDA->next != NULL)
{
addr = (unsigned long) cur_RDA->next;
u16 = addr >> 16;
l16 = addr;
addr=(unsigned long) u16 * 16 + l16;
n_RDA_addr=(unsigned short) addr;
cur_RDA->pkt_link=n_RDA_addr;
cur_RDA->status=0;
cur_RDA->byte_count=0;
cur_RDA->pkt_ptr=0;
cur_RDA->pkt_ptr1=0;
cur_RDA->seq_no=0;
cur_RDA->in_use=0xffff;
cur_RDA=cur_RDA->next;
}
/* last descriptor */
cur_RDA->pkt_link=0x0001; /* last desr. has EOL = 1 */
cur_RDA->in_use=0xffff;
lrda = cur_RDA;
}

```

**B. Initiate Transmission Routine**

```

/*****
*
Driver_send(). This routine, called by the upper level
software, gets the byte count, pointers to fragments and
the fragment sizes, enters these parameters into the TDA,
then initiates a transmission.

*****/
driver_send(ptr)
pktstruc *ptr      /*pointer to structure which gives
                    pkt_size, frag_count, frag_size */
{
    /* Fill out TDA */
    tda->pkt_size=packet_size;
    tda->frag_count=fragment_count;
    for (i=0; i<fragment_count; i++)
        Fill_fragment_ptr_size();

    /* Check packet length; if less than 46 bytes, add pad */
    Check_pkt_length();

    /* Get address of next TX descriptor to use */
    tda->link = get_next(); /* returns addr. of descr. */
    /* Set EOL.to 1. */
    tda->link |= 0x1;

    /* ISR will Set this flag to 1 */
    xmit_interrupt=0;

    /* Issue transmit command */
    REG_WRITE(card.crd_iobase+SONIC_cr*2, CMD_TXP);
}

```

TL/F/11140-28

## C. Interrupt Service Routine

```

/*****
Interrupt Service Routine

(For simplicity the code for recycling RBAs
has been removed.)
*****/

interrupt _sonic_isr()
{
    unsigned short imr, isr, mask;
    unsigned int status, byte_count;
    int oldinterrupts; long temp_ptr, ptr;

    mask=0;
    /* mask the imr */
    REG_WRITE(card.crd_iobase+SONIC_imr*2, mask);

    while (isr=REG_READ(card.crd_iobase+SONIC_isr*2)) {
        if (isr & ISR_PKTRX) {
            /* reset PKTRX bit */
            REG_WRITE(card.crd_iobase+SONIC_isr*2,ISR_PKTRX);

            /* Process receive packets */

            while (cur_rda->in_use == 0) {
                TotalRxPacketCount++;
                status = cur_rda->status;
                byte_count = cur_rda->byte_count;
                temp_ptr = cur_rda->ptr1;
                temp_ptr = temp_ptr<<16;
                ptr = temp_ptr | cur_rda->ptr0;
                /* Report packet to upper level software */
                packet_received(status,byte_count,ptr);

                /* Processing packets in order, when LPKT is 1,
                update the RWP register */
                if (cur_rda->status==RCR_LPKT) {
                    cur_rwp=cur_rwp->next;
                    /* advance rwp */
                    REG_WRITE(card.crd_iobase+SONIC_rwp*2,
                               cur_rwp->loc);
                }

                /* finish up receive */
                if (cur_rda->in_use == 0) {
                    cur_rda->in_use=0x0ffff;
                    cur_rda->pkt_link |= 0x1;
                    lrda->pkt_link &= 0x0ffffffe;
                    lrda=cur_rda;
                    cur_rda=cur_rda->next;
                }
            }

            /* check for RBE overflow (required) */
            isr=REG_READ(card.crd_iobase+SONIC_isr*2);
            if (isr & ISR_RBE) {
                /* Increment buffer overflow counter */
            }
        }
    }
}

```

TL/F/11140-29



```

        RRAExhaustCount++;
        /* reset RBE, this also causes the SONIC to read
           the RRA */
        REG_WRITE(card.crd_iobase+SONIC_isr*2,R_RBE);
    }

    /* check for RDE overflow (optional) */
    if (isr & ISR_RDE) {
        RDAExhaustCount++;
        REG_WRITE(card.crd_iobase+SONIC_isr*2, ISR_RDE);
    }
}

/* Process transmitted packets */
else if (isr & (ISR_TXER|ISR_TXND)) {
    xmit_interrupt=1;
    REG_WRITE(card.crd_iobase+SONIC_isr*2, ISR_TXER|ISR_TXDN);
    while (1) {
        if (tda->status & TCR_PTX) { /* Successful TX occurred */
            TotalTxPacketCount++;

            /* Post status of transmitted packet to
               upper level software */
            packet_tx(TX_status);
            /* Increment counters for net. management. */
            if (tda->status & TCR_DEF)
                DeferXmissionCount++;
            if (tda->status & TCR_NCRS)
                NoCRSCount++;
            if (tda->status & TCR_CRSL)
                CRSLostCount++;
            if (tda->status & TCR_OWC)
                OutOfWindowCollisionCount++;
            if (tda->status & TCR_PMB)
                PacketMonitorBadCount++;
        }
        /* TX abort condition occurred. CTDA register points to
           last descriptor attempted. */
        else {
            /* Increment counters for net. management. */
            if (tda->status & TCR_EXD)
                ExcessDeferalCount++;
            if (tda->status & TCR_EXC)
                ExcessCollisionsCount++;
            if (tda->status & TCR_FU)
                FIFOUnderRunCount++;
            if (tda->status & TCR_BCM) {
                ByteCountMismatchCount++;
                tda->pkt_size=Total_fragment_size(tda);
            }
            if (--RetryCounter == 0)
                HardTransmitErrorCount++;

            /* Post status of transmitted packet to
               upper level software that packet was undeliverable
            */
            packet_tx(TX_status);
        }
        else {
            /* resend the same packet again up to RetryCounter */
            REG_WRITE(card.crd_iobase+SONIC_cr*2, CMD_TXP);
        }
    }
}

```

TL/F/11140-30

```
    }  
    /* look for last descriptor in TX list */  
    if (tda->link & 0x1)  
        break;  
    else  
        tda=tda->next;  
    }  
}  
pic_eoi(card.crd_interrupt);  
REG_WRITE(card.crd_iobase+SONIC_imr*2, card.crd_intmask);  
}
```

TL/F/11140-31

## D. Diagnostic Routine

```

sonic_diag ()
{
    int oldinterrupt;
    struct aclock *clk, *alock_alarm();
    long timeout=0;
    unsigned short temp, ul6, l16, addr;
    unsigned long laddr;
    extern int timeout_func();
    short result;

    /* Before loopback test can commence SONIC needs to be
       initialized */

    /* check BNC cable connection: If transmission does not
       finish after specified time period (~1sec), the BNC
       connector is not connected. If excessive collisions
       occur, the cable is not terminated */

    clk=aclock_alarm(50,50,timeout_func, &timeout);
    REG_WRITE(card.crd_iobase+SONIC_isr*2, 0xffff);
    /* Get the 1st tda */
    laddr=(unsigned long) tda1;
    ul6=laddr >> 16;
    l16=laddr;
    laddr=(unsigned long)ul6 * 16 + l16;
    addr=(unsigned short) laddr;
    REG_WRITE(card.crd_iobase+SONIC_ctda*2, addr);
    tda1->link=0x0001;

    /* Issue transmit command */
    REG_WRITE(card.crd_iobase+SONIC_cr*2, CMD_TXP);
    for (timeout_value=0; timeout_value < 2; ) {
        temp=REG_READ(card.crd_iobase+SONIC_isr*2);
        if (temp & (ISR_TXDN | ISR_TXER))
            break;
    }
    clock_kill(clk);
    if (timeout_value) {
        check_cable=2;          /*Timeout occurred, BNC not connected*/
        goto final;
    }
    else if (tda1->status & TCR_EXC) {
        check_cable = 3;
        goto final;          /* Exc. Coll. occurred, cable not
                               terminated */
    }
    else
        check_cable = 1;

    /* MAC loopback */
    laddr = (unsigned long) F_RDA;
    ul6=laddr >> 16;
    l16=laddr;
    laddr = (unsigned long)ul6 * 16 + l16;
    addr = (unsigned short) laddr;
    REG_WRITE(card.crd_iobase+SONIC_crda*2, addr);
    mac_loopback=loopback(0x0200);

```

TL/F/11140-32

```

    if (mac_loopback != 1)
        goto final;

    /* ENDEC loopback */
    laddr = (unsigned long) F_RDA;
    u16=laddr >> 16;
    l16=laddr;
    laddr = (unsigned long)u16 * 16 + l16;
    addr = (unsigned short) laddr;
    REG_WRITE(card.crd_iobase+SONIC_crda*2, addr);
    endec_loopback=loopback(0x0400);
    if (endec_loopback != 1)
        goto final;

    /* transceiver loopback */
    laddr = (unsigned long) F_RDA;
    u16=laddr >> 16;
    l16=laddr;
    laddr = (unsigned long)u16 * 16 + l16;
    addr = (unsigned short) laddr;
    REG_WRITE(card.crd_iobase+SONIC_crda*2, addr);
    trans_loopback=loopback(0x0600);
    if (trans_loopback != 1)
        goto final;

    return(ok);

final:
    return(error);    /* one of the loopback test failed/*
}

/* This routine is to perform the loopback tests */

loopback( rcr_mode)
unsigned short  rcr_mode;
{
    struct aclock *clk;
    unsigned short temp, u16, l16, addr, rcr_value;
    unsigned long laddr;
    long timeout=0;
    short i;
    struct aphys *phys;

    /* Set up the clock to measure timeout */
    clk=aclock_alarm(50,50,timeout_func, &timeout);
    REG_WRITE(card.crd_iobase+SONIC_isr*2, 0xffff);
    /* Get the 1st tda */
    laddr=(unsigned long) tda1;
    u16=laddr >> 16;
    l16=laddr;
    laddr=(unsigned long)u16 * 16 + l16;
    addr=(unsigned short) laddr;
    tda1->link=0x0001;
    rcr_value=rcr_mode|0x3800;
    REG_WRITE(card.crd_iobase+SONIC_rcr*2, rcr_value);

    /* Out of reset mode */
    REG_WRITE(card.crd_iobase+SONIC_cr*2, 0);

```

```
REG_WRITE(card.crd_iobase+SONIC_ctda*2, addr);
REG_WRITE(card.crd_iobase+SONIC_cr*2, CMD_RXEN);
/* Issue transmit command */
REG_WRITE(card.crd_iobase+SONIC_cr*2, CMD_TXP);
for (timeout_value=0; timeout_value < 2; ) {
    temp=REG_READ(card.crd_iobase+SONIC_isr*2);
    if (temp & (ISR_TXDN | ISR_TXER))
        break;
}
clock_kill(clk);
if (timeout_value)
    return(2); /* timeout error */
else if (tdal->status & TCR_PTX) {
    if (F_RDA->status & RCR_LBK) {
        F_RDA->in_use=0xffff;
        return(1); /* good TX and RX status */
    } /* loopback OK */
    else {
        F_RDA->in_use=0xffff;
        return(3); /* Bad RX status error */
    }
}
else {
    F_RDA->in_use=0xffff;
    return(4); /* Bad TX status error */
}
}
```

TL/F/11140-34

# Determining Arbitration and Threshold Levels in a SONIC™ Based MicroChannel® Adapter

National Semiconductor  
Application Note 747  
Bill Carlson, FAE



## ABSTRACT

*With the number of bus master adapter boards increasing in MicroChannel based systems, many issues arise. This is especially true regarding Bus Master Ethernet LAN controllers such as the DP839EB-MCS. As such, the entire MCA environment needs to be considered so that critical settings for arbitration levels, threshold levels, and fairness options can be chosen. This paper describes these issues as they relate to National Semiconductor's DP839EB-MCS 32-bit Ethernet LAN controller board, which utilizes the DP83932 (SONIC).*

*The major issues include bus latency, bus efficiency and the contributing factors affecting these critical system level parameters. Factors such as bus occupancy times, DRAM refresh rates, floppy controller accesses, CPU accesses, mass storage transfer rates, latency tolerances, and priority levels all contribute to latency and efficiency. Within this environment, the high performance levels of the SONIC are achieved, even in worst-case scenarios in heavily loaded file servers with multiple bus masters.*

*It is also important to note that many of the basic concepts and considerations required in this application will also apply to other buses, although the detailed analysis will differ.*

## OVERVIEW

The DP83932 (SONIC) is a high performance, 32-bit, bus mastering Ethernet controller designed for a wide variety of applications. These applications include motherboards, routers, bridges and gateways, buffered and intelligent adapter boards, and bus master adapter boards. In each of these applications, determining the optimum thresholds and arbitration levels are key parameters to choose to ensure optimum performance. In determining these parameters, the anticipated system configuration needs to be understood. Specifically, the number and type of bus mastering devices in a system needs to be determined. Once these bus masters have been identified, the device thresholds and board arbitration levels can be determined.

Determining the anticipated number and type of bus masters directly affects a bus specification known as *Bus Latency*. Bus latency is defined as the time between when a bus master requests the bus to when it actually gets it.

Bus latency is a critical systems level specification because if it is too long, a bus master who doesn't get the bus when it needs it could suffer performance degradations or even more severe conditions such as a lost Ethernet packet or missed "sector" in a streaming tape drive. As such the Ethernet controller subsystem needs to have enough tolerance to handle large latencies to guarantee it's access to the bus and avoid this missed packet condition. The SONIC was specifically designed to perform in these applications.

By having a high speed, 66 MB/s, DMA host interface the SONIC maximizes bus bandwidth and minimizes time on the bus. Coupled with two efficient, 32 byte receive and transmit FIFOs, the SONIC will tolerate most latencies found in many applications.

Determining bus latencies is easy in many applications. Bridges and gateways, motherboards, intelligent and/or buffered adapter boards are systems in which the anticipated bus masters are known. In these systems it would be common to have the host CPU, a DMA controller, and peripheral devices (SCSI, FDDI, . . . ) all known by the system designer before the product is shipped out the door.

It is the designer who has to design a bus master adapter board or motherboard for a target bus (be in MicroChannel, EISA, VME, etc.) with expansion slots who has a tougher problem. He doesn't know what the end system configuration will be so he has to design to what is anticipated to be a worst case system configuration. The adapter board designer's customers would be the systems integrators who need to make sure that his board is designed properly so it will operate in fully loaded systems and still attain the high performance that he expects from this type of bus-mastering device.

Towards this end, this paper is written to assist the SONIC adapter board designer in choosing the correct arbitration and threshold levels for an IBM PS/2 Model 80 application, most probably operating as a file server having multiple LAN and mass storage devices on the MCA bus. For designers of other systems, this paper should help in understanding many of the issues that arise in a bus master LAN environment.

Before discussing this, a few MCA specifics need to be addressed. First off is the arbitration scheme. There can be up to 8 bus master expansion boards on the Model 80 MCA bus, including 8 DMA channels, the system CPU, refresh, and NMI which are on the system motherboard. Most have their own arbitration level as programmed via a POS register. When a device wants ownership of the bus, it asserts the PREEMPT\* signal and will then monitor the ARB/GNT\* signal, and when high (as controlled by the central arbitration logic on the system board) will place it's arbitration vector on the bus. If it's vector has the highest value, it wins the bus, ARB/GNT\* goes low, PREEMPT\* is de-asserted, and it can now do data transfers. If other devices want the bus they can asynchronously assert PREEMPT\*. The first device has 7.8  $\mu$ s to get off the bus and then all requesting devices, including the first if it wants to, compete for the bus and the arbitration process starts over again. When determining system characteristics, this 7.8  $\mu$ s is often used as it dictates the maximum amount of time that a device can own the bus if others are requesting it.

Another aspect of the MCA architecture is a feature called Fairness. Fairness allows all devices access to the bus in a round-robin fashion as determined by pre-assigned priority levels. Carefully choosing which devices are fair or not allows proper performance levels for the various devices on the bus. If fairness is enabled for a device and it currently owns the bus and another device(s) wants it, it will wait to re-arbitrate until all other requesting devices have had a chance on the bus themselves (this is noticed by the absence of an active PREEMPT\* signal). In this way no device will hog the bus and prevent others from accessing it. If fairness is disabled for a device, it will arbitrate for the bus any chance a valid arbitration cycle is available, regardless whether other devices are waiting to arbitrate also. Even with fairness enabled, the winner of the bus still needs the highest arbitration level, however, properly setting the fairness option will determine who will do the arbitrating.

In determining the arbitration levels and thresholds the designer of the SONIC bus master adapter board needs to account for a worst case bus situations. This would most likely be a high performance file server with multiple adapter boards. These could include an ESDI disk controller, an SCSI controller for additional disk and tape backup facilities and from 1 to 4 LAN boards to handle a heavily loaded network. Other anticipated bus master boards could also be included in this scenario (e.g., FDDI) but our discussion will be limited to the aforementioned configuration. (This is indeed a worst case scenario. A more typical case for a file server would have 1 or 2 LAN boards and both a SCSI and ESDI controller).

To summarize our worst case scenario for this analysis, we will assume the MicroChannel PS/2 has these adapter boards installed:

- 4 SONIC Bus Master Adapter Boards
- 1 Bus Master SCSI Controller
- 1 Bus Master ESDI Controller

#### DETERMINING ARBITRATION LEVELS AND THE FAIRNESS OPTION

When determining these it must be understood that the mass storage devices and the LAN controllers have different goals when it comes to bus utilization. The mass storage devices will have large blocks of data to transfer that are typically already stored in a local buffer on the adapter board or on the drive itself. All ESDI disk controllers have a local buffer, some with megabytes of storage. Most SCSI host adapters have buffering as well, although a trend is to use a bus-mastering SCSI controller IC that can gain the bus similar to the way the SONIC does. These don't have local buffering outside of their internal FIFO, but have the data storage on the disk drive itself. The main priority for the storage devices is to transfer as much data as possible for as long as it has the bus. Of second priority is latency toleration. These devices can wait a reasonable amount of time before they get the bus. Because they already have a large amount of data buffered, no data should be lost if it isn't granted the bus immediately. However, when it does get the bus, it needs to transfer as much as possible.

The Bus Master LAN controllers, on the other hand, need to have quicker access than the mass storage devices and within their latency period. This is especially true when receiving a packet, for to get a FIFO overrun error would cause upper protocol layers to initiate long and time con-

suming recovery procedures. Once they are on the bus, however, they are on for a relatively short period of time. This is due to the fast 20 MB/s MCA transfer rate and the smaller amount of data that is to be transferred at one time. (A disk or tape cache can have many Kbytes available for transfer, the 32 byte FIFO will transfer at the most that amount.)

With this in mind, the LAN controllers should be configured to have near immediate access to the bus. As such, each should be set to have a priority level higher than the storage devices. Thus whenever an arbitration takes place, a LAN controller should always participate and win so it can attain bus ownership as soon as possible. The setting of the fairness option should also be chosen to allow the LAN boards immediate bus access. If all devices had enabled the fairness option it is possible for the LAN board to be off the bus for a longer period of time than its latency tolerance allows, for example as shown in Table I.

**TABLE I. Possible (but Not Optimum) Priority Settings for Adapters, but Not the Optimum Solution**

Device	Priority	Fairness
LAN0	0	Yes
LAN1	1	Yes
LAN2	3	Yes
LAN3	4	Yes
SCSI	6	Yes
ESDI	7	Yes

In this scenario all devices have fairness enabled and the LAN boards have the higher priority. If a LAN board is awaiting arbitration it will win vs. the ESDI and SCSI boards. However, since fairness is enabled for the LAN boards it means that they must defer arbitrating until all other devices have been on the bus. These boards should participate in every arbitration cycle and by enabling fairness for them, this is prevented. Specifically in this example, the SCSI and ESDI boards will be on the bus consecutively for 7.8  $\mu$ s each (for 16.2  $\mu$ s total, including arbitration time) and the LAN boards would miss the intermediary arbitration cycle; this might exceed the boards latency toleration. By disabling fairness on the LAN boards, each is guaranteed to participate in every arbitration cycle and not have to wait for other device's arbitrations and bus occupancy times. Because of this and their higher priority levels, a LAN board will **always** arbitrate and win when an arbitration cycle occurs. We now have this:

**TABLE II. Priority Settings for Adapters with Correct Fairness Setting**

Device	Priority	Fairness
LAN0	0	No
LAN1	1	No
LAN2	3	No
LAN3	4	No
SCSI	6	Yes
ESDI	7	Yes

What about the storage devices? Fairness should be enabled for them. Due to the large amounts of data available for them to transfer in their respective caches, they will always have a need to own the bus and so they will always be requesting it. If fairness were disabled, the higher priority device (the SCSI controller in this case) would hog the bus and prevent the ESDI controller from accessing it. Thus fairness should be enabled for them.

To summarize, the above configuration will give each LAN board immediate access to the bus. The SCSI and ESDI boards would each have accessibility to the bus and although delayed due to the higher priority LAN boards, their latency tolerances are much higher and would incur only a minor, yet expected loss in bus acquisition time. The settings for the DMA slave ESDI controller that is configured with the Model 80, does indeed default to these settings. Fairness is enabled for it and it occupies DMA channel 7, the lowest priority DMA Channel.

The following *Figure 1* illustrates the sequence of events in a fully loaded, extreme worst case situation by properly setting the arbitration levels and fairness. Other devices such as refresh and the floppy controller will be included later when FIFO thresholds are discussed.

It should be remembered that the system CPU, the floppy controller, refresh, and other devices will be on the bus as well. These, along with the adapter boards all contribute to bus latency. Because of this latency the SONIC's FIFO threshold must be set properly to tolerate the expected latencies and avoid overrun/underrun errors. When set properly the SONIC will achieve the high performance the designer wants and the system's integrator expects.

#### DETERMINING THRESHOLD LEVELS

The FIFO threshold is an option that is programmed in the SONIC's Data Configuration Register and both the receive and transmit FIFOs can be programmed for different values. What is the FIFO threshold? The threshold is simply the point in time that the DMA engine requests the bus after a certain amount of data has filled the FIFOs. For example, a threshold of 1 long word for the receive FIFO would mean that after 4 received bytes from the network have filled the receive FIFO the DMA engine will request the bus. For the transmit FIFO, a threshold of 4 long words would cause the DMA engine to request the bus when the number of bytes in the FIFO falls below 16.

When determining the threshold levels, we need to first explore the specific latencies expected in our worst case scenario. The latency calculation is done by adding together the bus occupancy times of the various bus masters, their

priority levels, and the fairness option. We will assume the following:

- All adapter boards have 32-bit MCA bus master interfaces
- The SONIC board transfer rate will be at 250 ns (although MCA will operate @ 200 ns and the SONIC can do synchronous transfers on other buses @ 100 ns)
- Arbitration time will be 300 ns (0.3  $\mu$ s)
- EMPTY/FILL Mode is enabled for FIFO buffering
- The Floppy controller will request service from DMA Channel 2 every 12  $\mu$ s and will remain on the bus for 500 ns.
- Refresh occurs every 15.1  $\mu$ s and inserts itself in the middle of an arbitration cycle, extending it 200 ns for a total arbitration time of 500 ns.

In this example we will assume that the SCSI controller just got on the bus and then immediately afterwards all four LAN boards and the ESDI controller request the bus by asserting PREEMPT\*. This example takes a worst case latency and will show how the chosen threshold and arbitration levels and fairness options will guarantee proper system performance by showing how all four LAN boards will be able to access the MCA bus. When these devices request the bus it is to be understood that their FIFO thresholds have been reached. The LAN controllers will be buffering a received packet, a very critical bus access.

What should the threshold levels be for the 4 LAN controllers? Choosing the proper threshold involves trade-offs between a number of systems level specifications. By having a low threshold, maximum latency is assured. However, fewer bytes will transfer so the arbitration percentage will be higher, reducing efficiency. Also, the controller will request the bus more often causing bursty traffic across the bus. A larger threshold on the other hand, solves these problems at the expense of lower bus latency tolerance. In light of this, the thresholds of LAN0:1 should be higher than LAN2:3. LAN0:1 won't see larger latencies due to their higher priorities. However, they shouldn't request the bus again before LAN2:3 get a chance, increasing the latency they already incur. LAN2:3, however, need to tolerate longer latencies than LAN0:1 because, due to their priorities, they will be off the bus for longer periods of time. They will request the bus sooner and more often, however, this shouldn't impact system performance due to the short bus duration. By choosing a threshold of 16 bytes for LAN0:1 and 8 bytes for LAN2:3, as summarized in Table III, a good balance between these issues is achieved.

SCSI	LAN0	LAN1	LAN2	LAN3	ESDI	LAN0	LAN1	LAN2	LAN3	SCSI
------	------	------	------	------	------	------	------	------	------	------

FIGURE 1. Bus Ownership in Example PS/2 Under Worst Case Bus Request



Table III shows the arbitration bus priority assignments that show proper settings for the IBM PS/2 Model 80 devices. It should be remembered that these device assignments are determined by the MCA specification. Some of the assignments are pre-set, while others can be occupied by installable adapter boards. For example, refresh and NMI are pre-set to arbitration levels -2 and -1. The Floppy controller occupies DMA channel 2. The other DMA channels are available for adapter boards.

**TABLE III. Arbitration, Fairness, and FIFO Threshold Settings**

Device	Priority	Fairness	Threshold	Latency	Latency $\mu\text{s}$
Refresh	-2				
NMI	-1				
LAN0	0	No	16 Bytes	16 Bytes	12.8
LAN1	1	No	16	16	12.8
Floppy	2				
LAN2	3	No	8	24	19.2
LAN3	4	No	8	24	19.2
Available (Note 1)	5				
SCSI	6	Yes			
ESDI (Note 2)	7	Yes			
Available	8-E				
CPU	F				

**Note 1:** An IBM ST-506 disk controller will default to an arbitration level of 5 with fairness enabled.

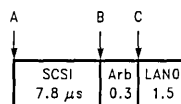
**Note 2:** An IBM ESDI controller will default to arbitration level of 7 with fairness enabled.

Devices 8-E are available for bus masters. In our example, DMA channels 0, 1, 3, and 4 are masked out and are used to hold the bus mastering LAN controllers. The bus master SCSI host adapter is put at ARB 6 with DMA channel 6 masked out. A standard PS/2 Model 80 comes with an ESDI disk controller operating as a DMA slave at ARB 7. This is the default setting for this controller. Because of this, the LAN designer doesn't have to worry about the arbitration level and fairness options for this controller. It can be assumed that the SCSI host adapter will be configured in the same way: with a low priority and with fairness enabled. In our example we have assumed a bus mastering ESDI controller; however, the standard one is a DMA slave device. For our discussion, though, we will assume it is a bus master for clarity's sake.

Once the arbitration levels and thresholds are determined for the LAN boards, they must be set when installed. IBM automatically sets the default values for the ESDI controller, but what about the LAN boards. How should they be set? Does the end user have to be aware of all these issues just to install a board? A simple solution would be for the driver to call a BIOS routine that would poll all the MCA slots to determine how many LAN boards are installed. The driver would then set the threshold and arbitration levels appropriately for each board. Using this method the user would be

far removed from the details of these specifics and a smooth installation would be insured.

At point "A" in Figure 2 below, LAN0:3 and the ESDI controller request the bus. At point "B", 7.8  $\mu\text{s}$  later the SCSI controller removes itself and an arbitration cycle begins with the other devices participating. It should be noted that if the bus-mastering SCSI controller IC is in the middle of a block transfer when it gets off, it will need to tell the target so it won't request more data transfers of it and the system any more. It does this by simply refusing to issue more acknowledgements to the target after the REQ/ACK offset has been met (in synchronous mode). In this way the target won't be requesting the initiator until it has access to the system bus again. The effect is that the SCSI controller can be off the bus even during the middle of a block transfer. After the arbitration following this SCSI transfer, LAN0 will win due to it's higher priority. To determine system latency we will need to calculate the sum total of the occupancy times of all devices. If this latency is less than the maximum latency tolerance of all the LAN devices, proper bus access and performance levels can be expected. If not, FIFO overruns would occur, the situation we are trying to prevent and will show won't happen.



TL/F/11141-1

**FIGURE 2. Initial DMA Sequence**

With that, how long will LAN0 be on the bus? Since LAN0 didn't get the bus until point "C", 8.1  $\mu\text{s}$  later, and the controller has been programmed for EMPTY/FILL mode, it will transfer the sum of the number of bytes determined by the FIFO threshold and the number of bytes accumulated from the network since the request was made. Let's call the "threshold" transfer time  $T_T$  and the transfer time for the accumulated bytes  $T_A$ . We will call the number of accumulated bytes simply "#". Since our threshold for LAN0 is 16 bytes,  $T_T$  will be the time it takes to transfer 16 bytes.  $T_A$  will be the time it takes to transfer the number of bytes accumulated since the request was made (8.1  $\mu\text{s}$ ), as well as  $T_T$ . So we have:

$$T_{TOT} = T_T + T_A$$

$$T_T = 16 \text{ Bytes} \left( \frac{1 \text{ Transfer}}{4 \text{ Bytes}} \right) 0.25 \mu\text{s}/\text{Transfer} = 1.0 \mu\text{s}.$$

$$\# = (8.1 \mu\text{s} + 1.0 \mu\text{s}) / (0.8 \mu\text{s}/\text{Byte}) \\ = 11.375 \text{ Bytes Accumulated.}$$

8 bytes (two long words) will transfer with 3 bytes left in FIFO and 3 bits in serial/parallel converter. (The SONIC will transfer only long-word values to/from the FIFO).

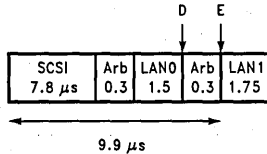
$$T_A = 8 \text{ Bytes} \left( \frac{1 \text{ Transfer}}{4 \text{ Bytes}} \right) 0.25 \mu\text{s}/\text{Transfer} = 0.5 \mu\text{s}.$$

$$T_{TOT} = 1.0 \mu\text{s} + 0.5 \mu\text{s} = 1.5 \mu\text{s}.$$

Therefore the total transfer time for LAN0 is 1.5  $\mu\text{s}$ . LAN0 will then request the bus again when it's FIFO threshold has been reached. Since there are 3 bytes left in FIFO and 3 bits in the serial/parallel converter,

$$T_{REQ} = (16 - 3 - \frac{3}{8} \text{ Bytes}) (0.8 \mu\text{s}/\text{Byte}) = 10.1 \mu\text{s}.$$

So LAN0 will request the bus 10.1  $\mu\text{s}$  later. It should be noticed that LAN0 (and LAN1 also) have a latency tolerance of 12.8  $\mu\text{s}$ . This latency is more than adequate for the current latency of 8.1  $\mu\text{s}$ .



TL/F/11141-2

**FIGURE 3. Initial Latency for LAN1 Card**

At point "D" LAN0 finished its transfer and LAN1:3 and the ESDI controller arbitrate with LAN1 winning due to its higher priority. Total bus occupancy for LAN1 will again be  $T_{TOT} = T_T + T_A$ .

$T_T = 1.0 \mu s$  (because of the 16 byte transfer as calculated above).

$$\# = \frac{9.9 \mu s + 1.0 \mu s}{0.8 \mu s/Byte} = 13.625 \text{ Bytes Accumulated.}$$

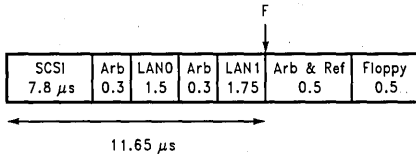
12 additional bytes (3 long words) will transfer with 1 byte remaining in the FIFO and 5 bits in serial/parallel converter.

$$T_A = 12 \text{ Bytes} \frac{0.25 \mu s}{4 \text{ Bytes}} = 0.75 \mu s$$

$$T_{TOT} = 1.0 \mu s + 0.75 \mu s = 1.75 \mu s.$$

Therefore LAN1 will own the bus for 1.75  $\mu s$ . Since LAN1's latency tolerance of 12.8  $\mu s$  is greater than the current latency of 9.9  $\mu s$ , it will be guaranteed access and no FIFO overruns will occur. LAN1 will then request the bus when its FIFO threshold has again been reached. Since there is 1 byte left in the FIFO and 5 bits in the serial/parallel converter, the request time will be:

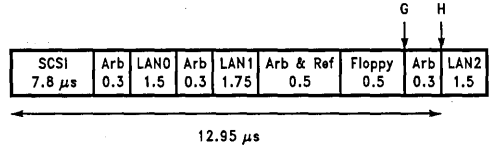
$$T_{REQ} = (16 - 1 - \frac{5}{8} \text{ Bytes}) (0.8 \mu s/Byte) = 11.5 \mu s$$



TL/F/11141-3

**FIGURE 4. Latency Till End of LAN1 Card Bus Occupancy Followed by Arbitration and Floppy Disk Access**

At point "F" the SCSI controller, LAN0 and LAN1 have had their turn on the bus. At this point another arbitration will take place. Since the system needs to refresh memory, we will put in a refresh cycle now. This refresh will extend the arbitration by 200 ns, to a total of 500 ns. We also need to account for a floppy controller access. It is important for the floppy controller to gain access to the bus because if one of its drives is a "floppy tape" and a byte was lost, the tape would have to stop, rewind, and re-read/write to that logical sector, taking a very bad performance hit. This situation needs to be prevented. We will assume that DMA channel 2 will win this arbitration and the floppy controller will transfer one byte, staying on the bus for approximately 500 ns. We now have:



TL/F/11141-4

**FIGURE 5. Bus Latency Time for LAN2 Card**

After the floppy access, LAN2:3 and the ESDI controller will arbitrate at point "G", with LAN2 winning and beginning to transfer at point "H". Since LAN2's latency tolerance is 19.2  $\mu s$  and 12.95  $\mu s$  is the current latency, there is 6.25  $\mu s$  of margin left to guarantee proper access. How long will LAN2 stay on the bus?

$$T_{TOT} = T_T + T_A.$$

$$T_T = 0.5 \mu s \text{ (for any 8 Byte Transfer)}$$

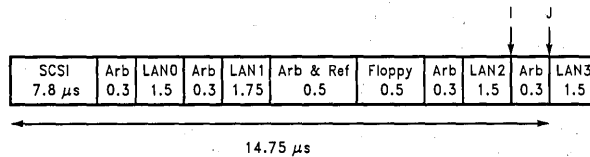
$$\# = (12.95 \mu s + 0.5 \mu s) (1 \text{ Byte}/0.8 \mu s) = 16.8125 \text{ Accumulated Bytes.}$$

The SONIC will then transfer the additional 16 bytes (4 long words) that were accumulated in the FIFO and keep the remaining 6.5 bits in the serial/parallel converter.

$T_A = 1.0 \mu s$  (from a previous calculation for a 16 byte transfer)

$$T_{TOT} = 0.5 \mu s + 1.0 \mu s = 1.5 \mu s.$$

LAN2 will then re-arbitrate when its FIFO has reached 8 bytes. This will be as shown in Figure 6.



TL/F/11141-5

**FIGURE 6. Bus Latency Time for LAN3 Card**

$$T_{REQ} = (8 - 6.5/8 \text{ Bytes}) (0.8 \mu\text{s}/\text{Bytes}) = 5.75 \mu\text{s later.}$$

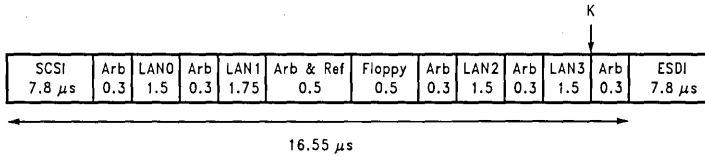


FIGURE 7. Total Bus Latency Time until Beginning of ESDI Drive Bus Access

TL/F/11141-6

At point "I" LAN2 is finished and LAN3 and the ESDI board will arbitrate with LAN3 winning. Since LAN3 has a latency tolerance of 19.2 μs and only 14.75 μs have occurred since LAN3 could have owned the bus, the latency margin of 4.45 μs is left over and a proper bus access has been guaranteed. LAN3 will then occupy the bus for:

$$T_{TOT} = T_T + T_A$$

$$T_T = 0.5 \mu\text{s (from before for an 8 byte threshold)}$$

$$\# = (14.75 \mu\text{s} + 0.5 \mu\text{s}) (1 \text{ Byte}/0.8 \mu\text{s})$$

$$= 19.0625 \text{ Accumulated Bytes.}$$

The SONIC will transfer 16 bytes (4 long words) with 3 bytes remaining in the FIFO and 0.5 bits in the serial to parallel converter.

$$T_A = 1.0 \mu\text{s for a 16 byte transfer so we have}$$

$$T_{TOT} = 0.5 \mu\text{s} + 1.0 \mu\text{s} = 1.5 \mu\text{s.}$$

LAN3 will then arbitrate again when its FIFO threshold of 8 bytes has been reached. This will be:

$$T_{REQ} = \left( \frac{8 - 3 - 0.5}{8} \right) (0.8 \mu\text{s}/\text{Byte}) = 3.95 \mu\text{s}$$

So LAN3 will request the bus again in 3.95 μs. At this point we have the following sequence of events:

At point "K", the ESDI controller will arbitrate and win and will stay on the bus for 7.8 μs. After winning the bus, the ESDI controller will deassert PREEMPT\*. The SCSI controller can now assert PREEMPT\* (because fairness has been enabled for it) to request the bus again since it has still more data to transfer.

In all of the previous illustrations we showed all devices and their respective occupancy times and their relative se-

quence. The following graph visually shows how long all devices will own the bus relative to each other. It is quite apparent that due to the SONIC's and MCA's high speed DMA, the LAN controllers are on for a minimal amount of time. Streaming Mode MCA adapters would be on for half the time.

In this example we have taken a worst case scenario by assuming all the LAN boards and the ESDI board will request the bus simultaneously at the very beginning of the SCSI transfer period. We have shown that even in this situation all devices have accessed the MCA bus without error and with plenty of latency margin left over. Table IV summarizes these results.

TABLE IV. Accrued Latency

Device	Accrued System Latency (μs)	Device Latency Tolerance (μs)	Latency Margin (μs)
SCSI	0	(Note)	
LAN0	8.1	12.8	4.7
LAN1	9.9	12.8	2.9
REFRESH	11.65		
FLOPPY	12.15	(Note)	
LAN2	12.95	19.2	6.25
LAN3	14.75	19.2	4.45
ESDI	16.55	(Note)	

Note: These latencies are particular to the device in question.

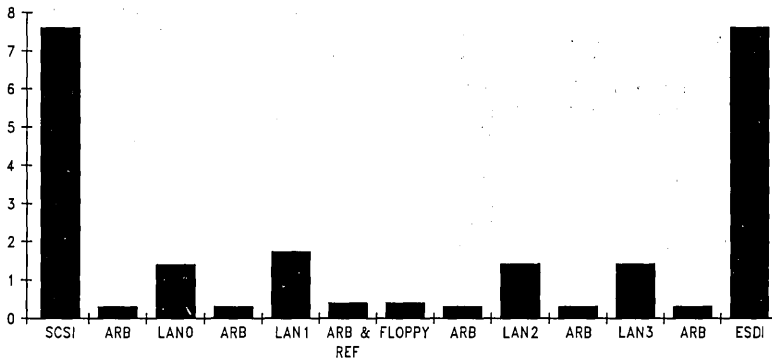


FIGURE 8. Individual Bus Usage Times for All Bus Masters, and Arbitration Cycles

TL/F/11141-8

Since we are basing our calculation on this simultaneous request, what will happen when these LAN boards arbitrate again? Will this worst case scenario happen again? Based on our previous calculations the LAN boards will request again at different times. The following diagram shows when the LAN boards will arbitrate once more:

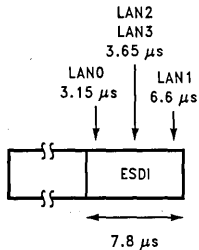


FIGURE 9

TL/F/11141-7

It can be seen now that starting with a worst case scenario as described above, the next set of LAN requests will be staggered apart throughout the ESDI transfer and our worst case scenario has all but disappeared, even after starting with it in the beginning. The LAN boards will still request and occupy the bus consecutively, however, they will now be on the bus for a shorter period of time. This is because the controller will get the bus sooner than in our worst case scenario; fewer bytes would have been accumulated in the FIFO since its threshold was reached hence a shorter transfer period. This means that other devices such as the CPU and mass storage controllers can have the bus sooner and occupy it longer than before. This equates to overall faster data throughput and more processing time for the CPU. It is up to the designer to determine when this worst case scenario will occur again, but it can be seen that the probabilities are exceptionally low that it will ever be repeated; however, if it did by properly setting arbitration and threshold levels and fairness options, the high performance of the SONIC can be readily achieved.

Since all devices have had a chance on the bus, what happens to the CPU during this worst case scenario? It has duties of its own such as protocol processing, updating descriptor lists, managing packets, etc. In the rare instance of this worst case scenario it wouldn't have immediate access to the bus. However, in nearly all the following accesses where the LAN accesses are staggered apart, there would be plenty of time for the CPU to access system memory.

One of the assumptions of this example is that no two consecutive transfers of 7.8 μs will occur in a row on the MCA bus when the LAN controllers are requesting it. The only way for this to happen was if there was a board which needed the bus immediately, and had a higher priority than the LAN boards and also would own the bus for a long period of time. However, a long bus occupancy time suggests a large buffer to hold all that data that is being transferred. A large

buffer means it can tolerate longer latencies which means it can be set to a lower priority level, which effectively means this situation is avoided. Thus the LAN boards can effectively remain at the highest priority level and not be potentially locked out due to multiple, consecutive, 7.8 μs transfers, which won't happen.

A concern throughout this analysis may be bus efficiency. Since the SONIC transfers just a few bytes at a time, it will request the bus often causing the arbitration time to be a significant portion of the transfer cycle. However, because of the Ethernet transfer rate of 1.25 MB/s these requests won't be often. When compared to the transfer times of the SCSI and ESDI boards, these arbitration times are not too significant (see Figure 8) and won't occupy much bus bandwidth. With these lower thresholds and bursty transfers, these inefficiencies become apparent. However, the SONIC more than compensates in other areas.

The 20 MB/s transfer rate of the DMA allows for minimal time on the bus. With Streaming Mode MicroChannel, the bus occupancy can be further lowered by having a 40 MB/s data rate. By keeping the FIFO down to 32 bytes, the buffering of runt packets is eliminated. A larger FIFO may buffer many of these unwanted packets in a heavily loaded network and wastes valuable bandwidth. Also, the SONIC's buffer management structure has been designed for simplicity and performance.

With much of the performance bottleneck happening in the upper protocol layers, a very fast and efficient driver becomes a necessity. The SONIC's register oriented buffer management scheme makes this possible. Updating descriptor lists is simple and doesn't take much processor overhead. It is very efficient.

The on-board CAM can hold up to 16 different physical and multicast addresses. This allows supporting multiple protocols at the MAC level. By assigning a different physical address to each of the different protocols supported by the file server, protocol filtering can be done at a very low level, where it is much more efficient. To implement this with a controller that supports only one physical address would necessitate it to enter promiscuous mode, meaning that it would have to buffer every packet on the network. This would be a very great waste of system bandwidth.

Another way to improve efficiency would be to tie multiple SONICs together while maintaining a single MCA bus interface. The MREQ\* and SMACK\* pins on the SONIC allow it to be a slave to other devices, even other SONICs. By tying multiple SONICs together, they could be time multiplexed into one MCA time slot; this would have the advantage of requiring only one arbitration cycle for multiple controllers. Not only would the efficiency go up but costs would come down as multiple SONICs would share just one bus interface. In short, the SONIC provides an optimal balance to achieve exceptional performance at all levels where system performance is measured.

# NM95C12 Applications in a PC-AT® Ethernet® Adapter

National Semiconductor  
Application Note 792  
Sean Long



## INTRODUCTION

This application describes a typical Ethernet adapter card designed to be plugged into a PC-AT expansion slot. The board is designed around the National Semiconductor DP83932 SONIC™ Network Controller device. This application note will detail the system design and focus on the functions performed by the NM95C12 EEPROM.

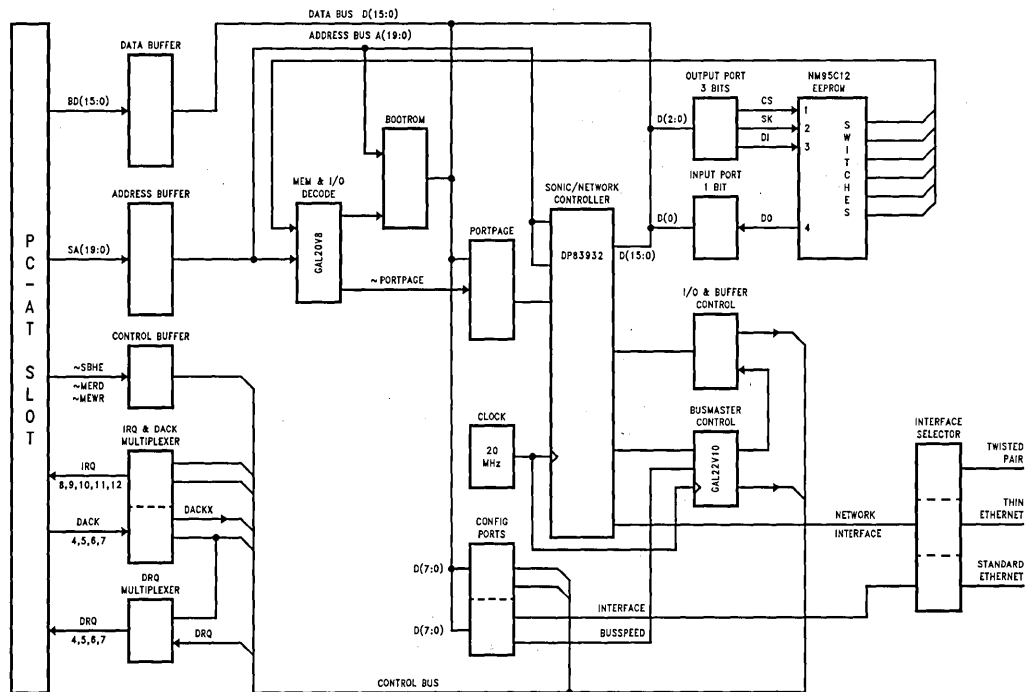
This application note assumes that the reader is familiar with the PC-AT architecture, the DP83932 device, the NM95C12 EEPROM and designing with GAL® Programmable Logic Devices (PLDs).

## SYSTEM DESCRIPTION

The network controller card has been designed to meet the following specifications:

- Designed around high performance 32-bit DP83932 Ethernet Controller
- 16-Bit bus master operation to give higher performance
- Fully software configurable (no jumpers or mechanical DIP switches)
- Extensive test and configuration capabilities
- Supports different media interfaces
- Bootrom option

The system block diagram is shown in *Figure 1*.



\*Denotes an active low signal.

FIGURE 1. System Block Diagram

TL/D/11265-1

## FUNCTIONAL DESCRIPTION OF THE BOARD

The system contains the following logical functions:

1. Network controller (DP83932)
2. Cable interfaces
3. Busmaster interface logic, including data and address buffers
4. EPROM option for remote boot loader

This system uses both the EEPROM locations and the switch logic terminals of the NM95C12 to perform various functions within the system as detailed below.

## FUNCTIONAL DESCRIPTION OF NM95C12 EEPROM

Use of the Switches:

The switch terminals of the NM95C12 EEPROM are used as part of the memory map address decoding and the I/O map decoding circuitry, feeding as inputs to a GAL20V8 which performs the address decoding logic from the system address inputs.

The NM95C12 switches control:

1. The base I/O address of the network controller board.
2. The base memory address of the bootrom EPROM option on the board.

## ADDRESS DECODING

The address decoding is controlled by a GAL20V8 PLD (refer to the 1990 National Semiconductor PLD Databook and Design Guide for further information) as shown in *Figure 2*. The inputs to the GAL20V8 are the system address lines, the memory and I/O control signals, and the switch termi-

nals from the NM95C12. The outputs from the GAL20V8 are the various chip select signals for the memory and I/O ports. The system address bus transmits the current address value and the  $M/\sim IO$  signal determines if a memory or I/O cycle is in progress.

Address lines A0-A19 allow up to 1 Meg (0-FFFFF) of memory to be addressed, while address lines A0-A15 allow up to 64K (0-FFFF) of I/O ports to be addressed. If the control signal  $M/\sim IO$  is logical "1" (high) then the processor is performing a memory cycle and if the  $M/\sim IO$  signal is logical "0" then an I/O cycle is in operation.

For a PC-AT various memory and I/O locations are reserved for standard functions such as system memory and I/O (refer to PC-AT documentation to determine which memory and I/O locations are free for add-in boards).

The switch outputs from the NM95C12 are connected as inputs to the GAL address decode logic and are used to determine the base memory and I/O locations for the add-in card. *Figure 2* shows the typical use of a GAL for address decoding.

The advantage of using a PLD for the address decoding is that it is an easy way to implement different address decode functions by logic equations. The logic equations can be implemented with a standard PLD design compiler such as OPAL™ from National Semiconductor or a third party software package such as ABEL™ from Data I/O. The PLD compiler will take the logic equations and convert them into the GAL fuse map which can be used for programming on a wide range of device programmers. A typical set of logic equations using National Semiconductors OPAL software package is shown in *Figure 3*.

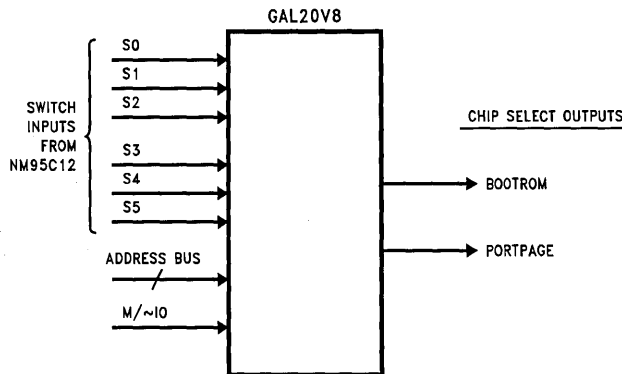


FIGURE 2. Address Decoding

TL/D/11265-2

```

BEGIN HEADER
TITLE   Address decoding for PC AT Ethernet adapter card
PATTERN Addr_Dec
REVISION Rev 0
AUTHOR   Dave Engineer
COMPANY  National Semiconductor
DATE     June 1991
Everything in the header command is copied directly into the JEDEC map as a comment field for
easy documentation
END HEADER

```

```

BEGIN DEFINITIONS
device G20V8;                { specify the device used }
inputs s0, s1, s2, s3, s4, s5;    { define the inputs }
inputs m_~io, a0, a1, a2, a3, a4, a5;
outputs (com) bootrom, portpage;    {define the outputs}
{ OPAL will perform automatic pin assignment }
set ioselect=[s2,s1,s0], memselect=[s5,s4,s3]; { define the switch sets }
set address=[a5,a4,a3,a2,a1,a0];
END DEFINITIONS

```

```

BEGIN EQUATIONS
{ " / " = logical NOT function (i.e. logical 0)
  " & " = logical AND function
  " + " = logical OR function }
{ if m_~io is logical 0, then decode switch set s2, s1, s0 and address lines for the various
base I/O locations.
  Refer to PC-AT system I/O address map before selecting free I/O ports, the decodes shown are
  for example only - change for specific applications as required. }
bootrom = /m_~io & ( (ioselect == 0) & (address == ↑h00)
  + (ioselect == 1) & (address == ↑h01)
  + (ioselect == 2) & (address == ↑h02)
  + (ioselect == 3) & (address == ↑h03)
  + (ioselect == 4) & (address == ↑h04)
  + (ioselect == 5) & (address == ↑h05)
  + (ioselect == 6) & (address == ↑h06));
{ if m_~io is logical 1, then decode switch set s5, s4, s3 and address line for the various
base memory locations.
  Refer to PC-AT system I/O address map before selecting free Memory locations, the decodes
  shown are for example purposes only - change for specific applications as required. }
portpage = m_~io & ( (memselect == 0) & (address == ↑h18)
  + (memselect == 1) & (address == ↑h20)
  + (memselect == 2) & (address == ↑h28)
  + (memselect == 3) & (address == ↑h30)
  + (memselect == 4) & (address == ↑h38);
END EQUATIONS

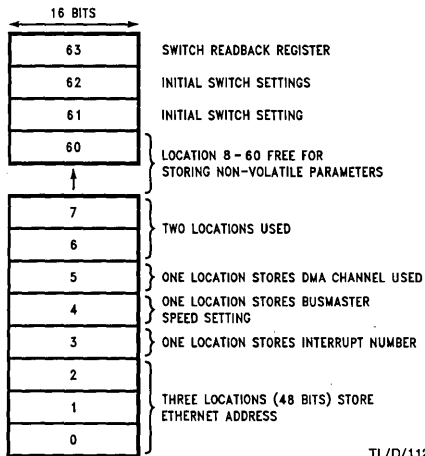
```

FIGURE 3. GAL® Logic Equations

#### USE OF THE NM95C12 EEPROM LOCATIONS

- Three locations are used to store the ethernet address of the card.
- One location is used to store the interrupt number and the DMA channel of the board.
- One location is used to store the busmaster speed setting of the card.
- Two locations are used to store information about the production flow of the board e.g.; the version number of the out-going inspection, and serialization program which stores a unique ethernet address in the EEPROM.
- There are also some EEPROM locations used to enable some special features in the network driver such as protocol, DMA priority, etc.

Figure 4 below shows the memory usage of the NM95C12.



**FIGURE 4. Memory Locations Used in NM95C12**

#### FUNCTIONAL DESCRIPTION OF THE SOFTWARE

The driver for the card can be supplied in two ways:

1. As a driver which is loaded from the disk.
2. As a bootrom which is located at the card.

The driver determines the base I/O address of the card. This is done by scanning the possible I/O map where the card can be located (seven possible locations) and testing if the NM95C12 EEPROM can be found.

The EEPROM is found if, after an address is shifted in the EEPROM, the DO output from the NM95C12 has become logical "zero". Then the CS pin will be disabled and there will be a check if the DO output pin will become high (this pin is pulled up with a 47K resistor).

When the software finds the base address of the card, it reads the locations which contain the DMA and IRQ number to use and programs these values into the corresponding output latches. These latches will enable and/or multiplex the corresponding DMA (DACKx, DRQx) and INT (IRQx) to the busmaster logic and interrupt logic.

The same operation is done for the busmaster speed, one location in the EEPROM determines the active low and high time for busmaster cycles, the output of this latch will go to the busmaster state machine (implemented in a GAL22V10).

The ethernet address will be read by the driver and copied to a private location in the driver data area for use with the network software.

The bootrom can be located at five locations in memory (controlled by the NM95C12 switch logic) and can be disabled if required.

#### CONCLUSION

This application has shown the many advantages of the NM95C12 EEPROM with DIP Switches. In this example the NM95C12 replaces the functions typically performed by a Bipolar PROM (store ethernet address), mechanical DIP switches/jumpers (select options), and general read/write logic (software testing of the hardware configuration). The use of the switch terminals as part of the address decode logic makes the address decode function more flexible and allows for software control.

The easy interfacing to the NM95C12 (just four pins) and the simple, but powerful instruction set allows the NM95C12 to give the system designer:

- Greater flexibility
- Fully software controllable and testable
- Greater reliability (no mechanical switches or jumpers)
- Reduced component count
- Lower component cost



# DP83916EB-AT: High Performance AT Compatible Bus Master Ethernet® Adapter Card

National Semiconductor  
Application Note 855  
Denise Troutman



## INTRODUCTION

The DP83916EB-AT board is a high performance 16-bit Ethernet adapter card designed for IBM® PC-AT®/compatible computer systems. It employs a unique bus master architecture which transfers packet data at rates up to 10 Megabytes/second to and from the PC-AT's system memory during Ethernet reception and transmission. Featuring the National Semiconductor DP83916 Systems-Oriented Network Interface Controller (SONIC™-16) and the PLX AT9010, the board functions as a bus master adapter card for implementing Ethernet nodes. Its design includes an interface which enables PC-AT managed-hub applications. Furthermore, it supports three types of media for functionality in IEEE 802.3 networks.

By using the DP83916 SONIC-16, the DP83916EB-AT board maximizes bus master performance over existing adapter cards. First, the SONIC-16's highly integrated design eliminates the need for I/O mapped or dual port adapter RAM designs. Second, the SONIC-16's bus master architecture facilitates writing and reading network data directly to and from main system memory. This architecture is supported by the SONIC-16's bus latency tolerance, its link-list buffer management scheme and its 24-bit memory addressing capability.

The PLX AT9010/AT9010B provides a compact, inexpensive and high performance AT bus interface for the DP83916EB-AT adapter card. It integrates much of the AT/SONIC-16 signal decoding and control logic. Because most options are selected by software drivers, the use of jumpers is reduced.

The DP83916EB-AT offers management interface logic to implement a managed hub when the board is coupled with the DP83950EB-AT RICKIT. This interface allows the SONIC-16 to emulate a RICTM on the Inter-RIC™/Management bus; hence, the SONIC-16 can receive packets containing network data and collision information and also transmit packets over this bus. By using hub management, the DP83916EB-AT makes network statistics available to a manager located anywhere on the network.

Finally, the adapter card offers multiple IEEE 802.3 cable media options: the DP8392 Coaxial Transceiver Interface for Thin Ethernet and an AUI connector for Thick Ethernet or Twisted-Pair.

Use of the DP83916EB-AT adapter card provides a low cost 15-chip bus master Ethernet node that supports three media options for IEEE 802.3 networks. SONIC-16 hub management is included by the addition of only five chips and the DP83950EB-AT RICKIT.

## FEATURES

- Efficient NSC DP83916 Systems-Oriented Network Interface Controller (SONIC-16)
- Highly integrated PLX AT9010/B for PC-AT bus master interface
- Inter-RIC/Management bus interface for optional connection to DP83950EB-AT RICKIT
- Selectable media interfaces: Thin Ethernet or AUI for Twisted-Pair and Thick Ethernet
- Optional EPROM for remote system boot
- Four programmable master data transfer speeds including 5, 6.7, 8, and 10 Megabytes/second
- Selectable interrupt lines, bus request channels, adapter card I/O addresses and EPROM memory addresses

## HARDWARE OVERVIEW

A block diagram for the DP83916EB-AT board is shown in *Figure 1*. The design can be broken into four sections: slave logic, master logic, hub management and media interface.

The slave logic facilitates the AT's CPU in accessing the SONIC-16's registers, the AT9010's registers, the Ethernet Address ID PROM and the network Boot EPROM (socket). Much of the slave circuitry decode (chip selects) and control logic is implemented in the AT9010. The slave devices are accessed in I/O and memory space.

The bus master logic assists the SONIC-16 in transferring data directly to and from the AT's system memory. It includes all signal translation and control logic required to access the bus, a large portion of which is integrated in the AT9010. The SONIC-16 uses the DMA controller to arbitrate between bus requestors for control of the bus and additional buffers for address and data buses.

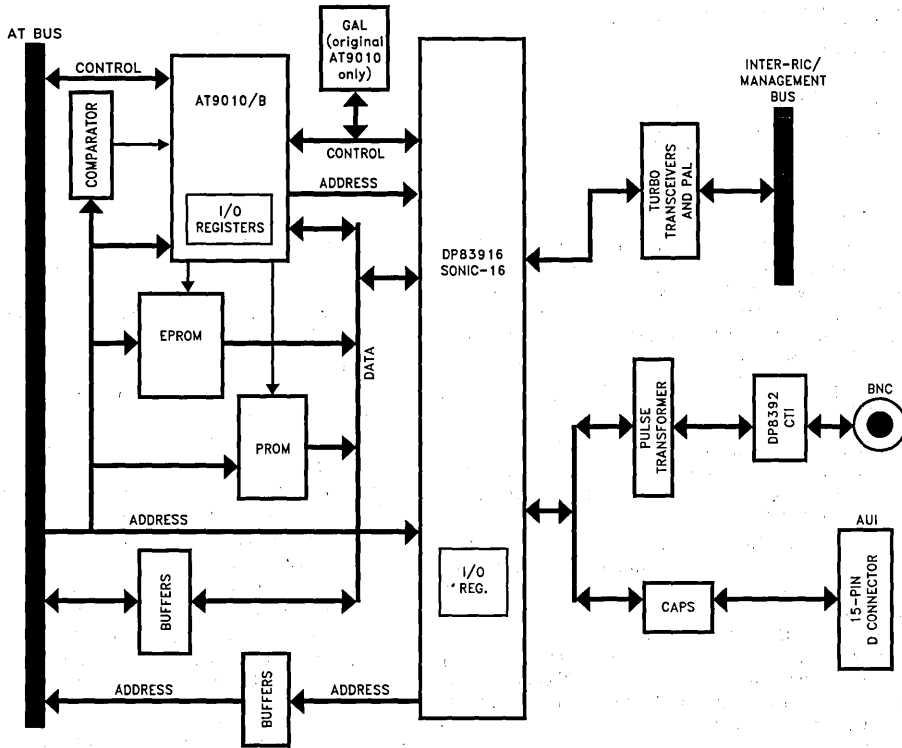


FIGURE 1. DP83916EB-AT Board Block Diagram

TL/F/11707-1

The hub management logic interfaces the Inter-RIC/Management (IR/M) bus and the SONIC-16. The IR/M bus enables the SONIC™ to gather network statistics for packets transmitted from a DP83950EB-AT RIC evaluation board, to receive control packets from remote nodes (managers) and to transmit packets to the network. The logic includes turbo transceivers (BTLs) for driving signals, a PAL® for IR/M bus arbitration signals and a flip-flop which provides the IR/M transmit clock.

The media interface connects the adapter card to one of three network media choices: Coax for 10BASE2 (Thin Ethernet) and AUI for 10BASE5 (Thick Ethernet) and 10BASE-T (Twisted-Pair).

Connection requirements for each choice will be described later.

The DP83916EB-AT supports two versions of the PLX interface chip for the SONIC-16 and the PC-AT platform: the original AT9010 and the AT9010B. If the board is populated with the AT9010, a GAL (U11) is required to fix bugs in the AT9010 chip. The bugs are corrected in the AT9010B and adapter cards containing this version of the chip **should not** have a GAL placed in the socket for U11. (Note, all subsequent references in this application note to "AT9010" apply to both versions of the chip unless specified otherwise.)

Since the GAL is not populated when the AT9010B is populated, signals driven from the AT9010B into the GAL will have no effect.

The appendices at the end of this application note provide the following information: Appendix I—PAL and GAL equations, Appendix II—Bill of Materials (BOM), Appendix III—AT9010/AT9010B Register Descriptions, Appendix IV—Signal Descriptions, Appendix V—DP83916EB-AT Card Layout, Appendix VI—Test Pin Layout, Appendix VII—Design Change Recommendation and Appendix VIII—Compatibility Testing. In addition, a detailed schematic for the DP83916EB-AT adapter card is also located at the end of this application note.

#### SLAVE LOGIC SECTION

During slave cycles, the AT's CPU accesses one of the four slave devices on the card: the SONIC-16's 64 internal registers, the Ethernet address PROM, the AT9010's five internal registers or the boot EPROM. Slave mode also includes card initialization.

The AT9010 provides chip select/control and signal conversion logic. Its configuration registers specify I/O address mapping and SONIC-16 register select information, EPROM memory address decoding and interrupt line specification.

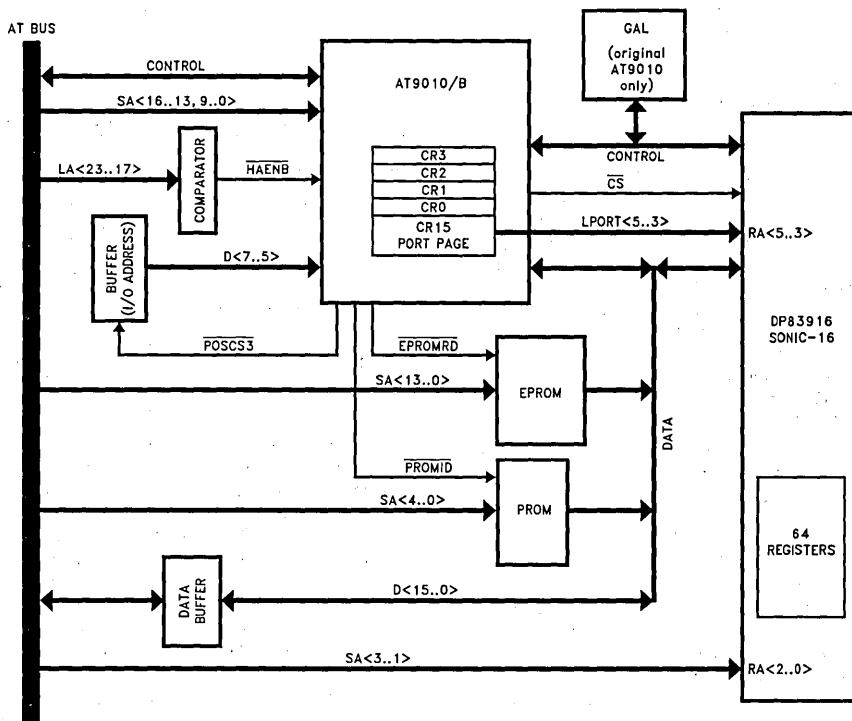


FIGURE 2. Slave Logic Block Diagram

TL/F/11707-2

**I/O MAPPING**

The SONIC-16 registers, AT9010 registers and Ethernet address PROM reside in the PC-AT's I/O space. The AT9010 offers a choice of seven 32-byte I/O blocks (shown in Table I) to place the card within I/O space. The user can map the board into any one of these blocks by selecting an unused portion of I/O space and positioning jumpers JP1-JP3 accordingly. A shorted jumper corresponds to a 0 and an open jumper corresponds to a 1. The values of these jumpers are enabled onto the data bus during power up and subsequently initialize the AT9010's Configuration Register 1, bits 7, 6, and 5.

TABLE I. I/O Space Mapping

JP1	JP2	JP3	CR1 Bits <7..5>	I/O Address (Hex)
0	0	0	000	100-11F
0	0	1	001	120-13F
0	1	0	010	140-15F
0	1	1	011	160-17F
1*	0*	0*	100	300-31F*
1	0	1	101	320-33F
1	1	0	110	340-35F
1	1	1	111	340-35F

\* DP83916EB-AT default setting  
0 = short (jumper on), 1 = open (jumper off)

The actual mapping of the SONIC-16 registers, AT9010 registers and SONIC-16 paging register into one of the 32-byte blocks of I/O space is shown in Figure 3. Bytes 2-5 are the AT9010's Configuration Registers 0-3, bytes 8-13 are the Ethernet address PROM, byte 15 is AT9010's Configuration Register 15, and bytes 16-32 are the SONIC-16's registers. All other bytes are reserved.

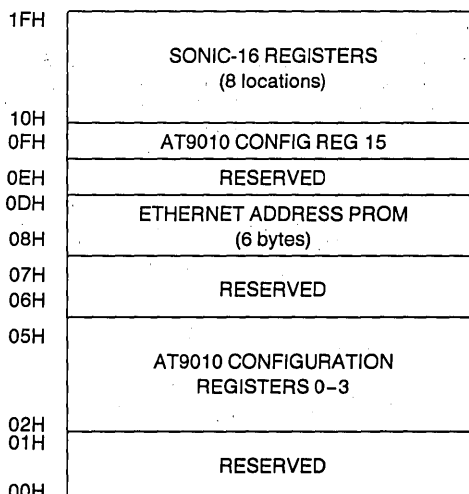


FIGURE 3. Card 32-Byte I/O Space Map

### SONIC-16 AND AT9010 REGISTER ACCESS

Due to limited PC-AT I/O space, only eight of the SONIC-16's 64 registers are accessible at any one time. The DP83916EB-AT accommodates this by partitioning the SONIC-16's registers into eight pages of eight registers (16-bit locations).

To access the registers, the CPU must drive a 6-bit register address. First, the CPU sets up the page number (0 to 7) by executing an 8-bit I/O write cycle of D<5..3> to the AT9010's Configuration Register 15, bits<5..3>. These data bits drive the three most significant SONIC-16 register address bits RA<5..3>. Then, the CPU executes a 16-bit I/O read or write cycle using the PC-AT's lower address lines SA<3..1> to drive the SONIC-16's address bits RA<2..0> which select the appropriate register. This completes the SONIC-16 register access. Note, most SONIC-16 register accesses actually require only one I/O cycle because the SONIC-16 registers which are accessed most often are located on Page 0.

The AT9010 configuration registers are 8 bits wide and are read or written through standard 8-bit I/O cycles.

### PROM AND EPROM MEMORY MAPPING

The PROM is a 32-byte register which holds a unique 6-byte Ethernet ID Address in offset locations 08H–0DH. It is read by 8-bit I/O read cycles.

The optional boot EPROM (socket), which resides in the PC's BIOS memory space, is also an 8-bit device. If used, the EPROM can be programmed with instructions that are scanned on power-up and enable a diskless workstation to boot up remotely from a network, then access a server.

The boot EPROM can be memory mapped above 640k within the first megabyte of memory. Specifically, the AT9010 places the EPROM in a section of memory within the 0C0000H to 0DFFFFH address range. As shown in *Figure 4*, the upper address decode bits for LA<23..17> are predetermined by the AT9010. To complete the base address, the decode bits for SA<16..13> must be programmed by the user in Configuration Register 2. The AT9010 can be configured for an 8k, 16k or 32k EPROM.

During EPROM memory accesses, address lines LA<23..17> are decoded to notify the AT9010 of EPROM activity. This decode is accomplished by the comparator shown in *Figure 2*.

The DP83916EB-AT design supports an 8k or 16k EPROM (socket); hence, the memory base address options are 8k or 16k sections of memory. Table II shows the decode for a 16k EPROM. Note, SA13 is driven directly into the EPROM; therefore, bit 2 of Configuration Register 2 is a don't care bit.

If the EPROM is used, Configuration Register 2, bits 7 and 6 must be set to 1 and 0 to enable an 8k EPROM or 0 and 1 to select a 16k EPROM. If the EPROM is not used, these bits must be set to 1s to disable the EPROM.

Comparator Decode							Configuration Register 2			
LA23	LA22	LA21	LA20	LA19	LA18	LA17	SA16	SA15	SA14	SA13
0	0	0	0	1	1	0	bit5	bit4	bit3	bit2

FIGURE 4. EPROM Address Bit Specification

TABLE II. EPROM Memory Base Address

CR2 Bits<5..2>				Base Address 16k EPROM (Hex)
bit5	bit4	bit3	bit2	
0	0	0	X	0C0000
0	0	1	X	0C4000
0	1	0	X	0C8000
0	1	1	X	0CC000
1	0	0	X	0D0000
1	0	1	X	0D4000
1	1	0	X	0D8000
1	1	1	X	0DC000

### INTERRUPT LINE SELECTION

As the SONIC-16 transmits and receives packets from the network, it generates interrupts (via an IRQx line from the AT9010) to the CPU. This results in several slave cycles which read the SONIC-16's Interrupt Status Register and service the interrupt. When the original AT9010 chip asserts an interrupt, it sets a non-latched interrupt indicator in bit 2 of Configuration Register 15 that is cleared when IRQx is cleared. (In the AT9010B, the interrupt indicator is latched; hence, it is maintained even if IRQx deasserts and is cleared by writing a 1 to CR15, bit 2.)

The DP83916EB-AT user can select one of four interrupt lines by programming the AT9010's Configuration Register 0. Table III presents the IRQx choices (lines 3, 4, 5 or 9) and indicates the necessary bits to select the appropriate line. In addition to choosing an IRQx line, the user must also program Configuration Register 0, bit 3 to mask (0) or unmask (1) the chosen interrupt upon assertion of INT from the SONIC-16.

TABLE III. Interrupt Line Selection

CR 0, Bits<2,1>	PLX Reference	IRQx Signal
00	0	3
01	1	4
10	2	5
11	3	9

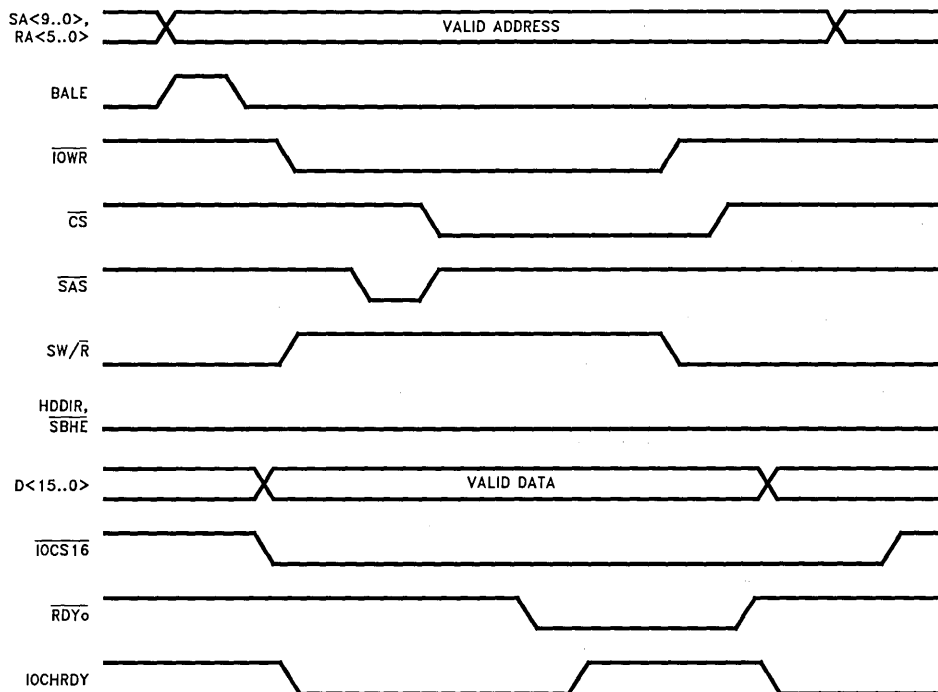


FIGURE 5. I/O Write Cycle to SONIC-16 Registers

TL/F/11707-3

## SLAVE CYCLES

The following section presents a basic description and timing diagrams for the signals generated by the AT bus, AT9010 and SONIC-16 when the DP83916EB-AT is in slave mode. A detailed description of the relevant signals associated with AT I/O cycles to the adapter card and AT memory accesses to the EPROM is located in Appendix IV at the end of this application note.

The DP83916EB-AT adapter card is designed so that on power-up, the  $\overline{\text{POSCS3}}$  pin will enable the I/O address of the adapter card onto the data bus. This address is latched into bits <7..5> of Configuration Register 1 and locates the card in I/O space. Software drivers must subsequently program all registers with the correct data for operation.

The AT bus initiates an I/O cycle by driving the address onto the bus, asserting BALE high and generating  $\overline{\text{IORD}}$  or  $\overline{\text{IOWR}}$ . The AT9010 then generates a chip select to the appropriate slave device.

The SONIC-16 registers, which are located in I/O space, can be written or read. Because they are 16 bits wide, a special signal  $\overline{\text{IOCS16}}$  is driven to the AT bus for the duration of the SONIC-16 I/O access. The AT9010 drives IOCHRDY low (not ready) after the address and  $\overline{\text{IORD/IOWR}}$  signals are asserted (and  $\overline{\text{MEMR/MEMW}}$  are not active). (Note, in this document, IOCHRDY will not be broken down into IOCHRDYBUS and IOCHRDYAT in order to give a clear explanation of the signal IOCHRDY's purpose. For an explanation of these signals, refer to the signal descriptions and GAL equations in the appendices and the schematic at the

end of this application note.) When the SONIC-16 has latched write data or driven valid read data, it generates  $\overline{\text{RDY0}}$  to terminate the cycle. At this time, the AT9010 asserts IOCHRDY high to the AT bus. An example of the signal timing for an I/O write cycle to SONIC-16 registers is shown in Figure 5.

Most I/O devices (like the DP83916EB-AT during slave mode) drive IOCHRDY low after the address and I/O command signal are asserted. However, some AT compatible machines use chip sets (from Chips and Technologies or VLSI Technologies) with modified timing characteristics whereby during 16-bit I/O cycles, the PC's bus controller samples IOCHRDY before  $\overline{\text{IORD}}$  or  $\overline{\text{IOWR}}$  is driven. This problem is detailed in the NSC document "PC-AT Design Considerations for the DP83902EB-AT".

To accommodate this early sampling problem, the AT9010B can be programmed to drive IOCHRDY low immediately upon SONIC-16 register address decode. The AT9010B then uses a gating signal to maintain IOCHRDY if the cycle is an I/O cycle or to stop asserting IOCHRDY if the cycle is a memory cycle. Specifically, if the AT9010B Configuration Register 3, bit 3 is set to 1, IOCHRDY will be driven low early, based on I/O address decode of a SONIC-16 register, then qualified with a gating signal. If CR3, bit 3 is set to 0, IOCHRDY will follow the AT standard: address decode of a SONIC-16 register and  $\overline{\text{IORD/IOWR}}$  and inactive  $\overline{\text{MEMR/MEMW}}$ . The early IOCHRDY feature should not be used if the PC I/O cycles are functioning properly. Furthermore, it is not supported by the original AT9010 chip.

The AT9010's registers can be written or read. The 32-byte PROM, however, can only be read. For either device, the I/O read cycles are finished once the data has been enabled onto the bus and read by the CPU. An I/O write cycle to an AT9010 register is completed once the data has been latched to the AT9010. IOCHRDY is not driven low during the 8-bit cycles to either of these devices.

The AT bus initiates a boot EPROM cycle by driving the address onto the bus, asserting BALE high and generating MEMR. Once the AT9010 has chip selected the EPROM, data is enabled onto the data bus and the memory cycle is complete. An EPROM read cycle is illustrated in *Figure 6*. IOCHRDY is not driven during EPROM cycles.

#### DP83916EB-AT REGISTER INITIALIZATION

Upon power-up, the DP83916EB-AT card pulses the card's I/O address into Configuration Register 1. Subsequently, software initializes Configuration Registers 0-3 and 15 for operation.

The original AT9010 is enabled by setting bit 0 of Configuration Register 0 to a 1. If this bit is a 0, the original AT9010 will not respond to any host bus access except hard reset. In the AT9010B, the card is enabled regardless of the state of this bit.

The AT9010B supports software reset. If Configuration Register 15, bit 7 is set to a 1, the AT9010B will reset all the

chip's functions to the default condition except Configuration Registers 0-3 and bits 3-5 and 7 of Configuration Register 15. The software must subsequently write a 0 to CR 15, bit 7 to clear this bit. Bit 7 of CR 15 is a reserved bit in the original AT9010. If it is set to 1, the card is lost in I/O space and can only be recovered by hard reset.

**Note:** The AT9010B corrects bugs in the AT9010 and offers additional features. The DP83916EB-AT card and demonstration software support both versions of the interface chip. Appendix III provides details regarding specific programming of all Configuration Registers.

#### MASTER LOGIC SECTION

During master mode, the AT's CPU allows the SONIC-16 to take over the system bus and access main memory directly. The SONIC-16 uses the DMA controller to assist in the bus arbitration process. In addition, it utilizes the AT9010's bus interface logic for requesting the bus and generating AT compatible read/write signals. A block diagram showing the master logic is presented in *Figure 7*.

#### DMA CONTROLLER CHANNEL SELECTION

For data transfer, PC-ATs utilize two 8237A DMA controllers with four channels each. Controller 1 contains channels 0-3 which support byte transfers and are typically reserved for diskette, SDLC, etc. It is cascaded with Controller 2 which contains channels 4-7 to support word transfers. While channel 4 is used to cascade Controller 2, channels 5-7 are usually spare.

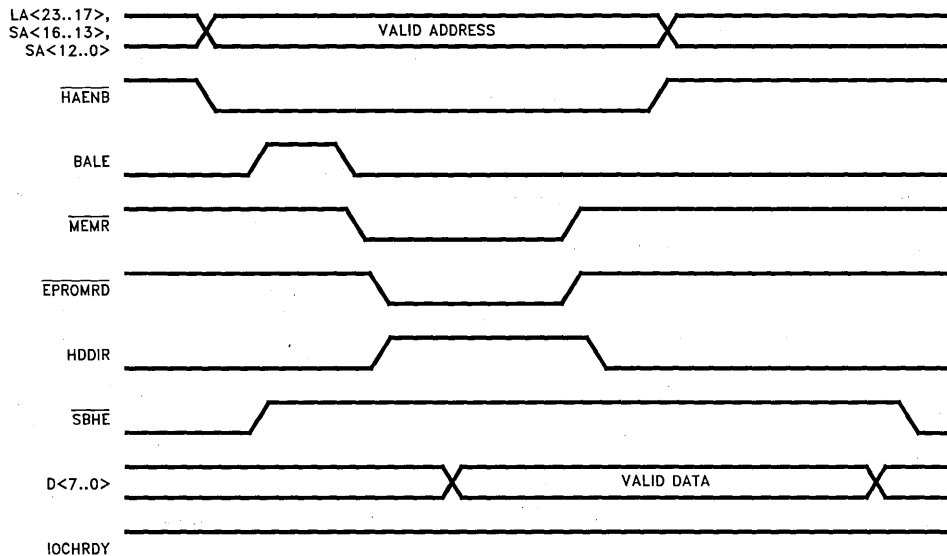


FIGURE 6. Memory Read Cycle to Boot EPROM

TL/F/11707-4

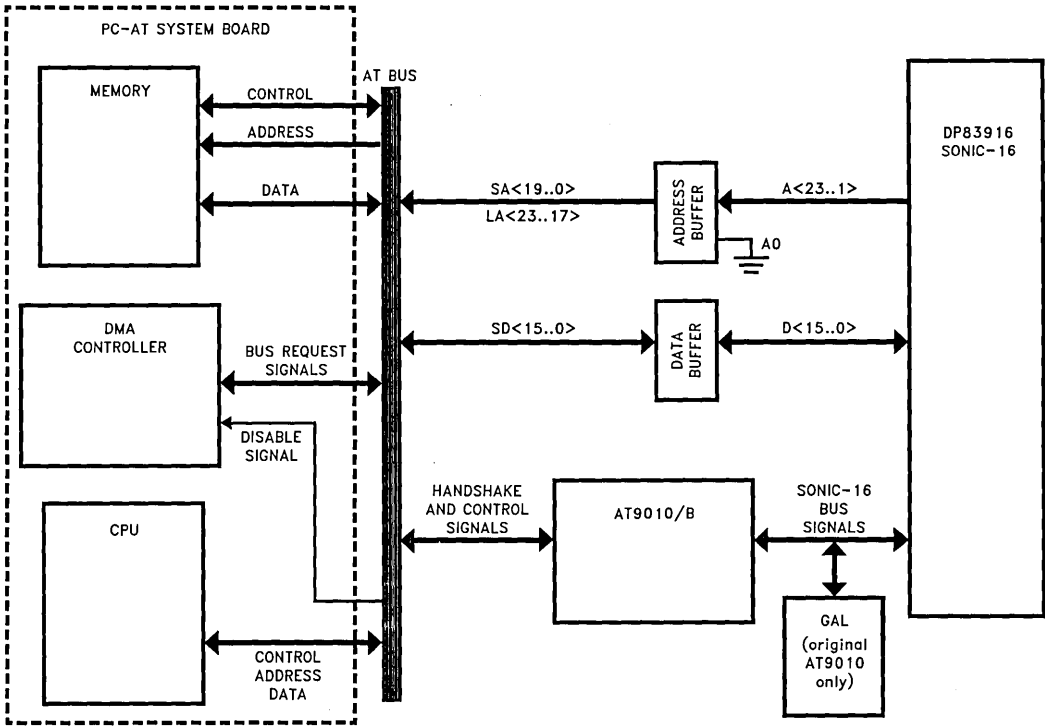


FIGURE 7. Master Logic Block Diagram

TL/F/11707-5

The user must select one of the spare DMA channels. By programming the AT9010's bits 0 and 1 of Configuration Register 3, the user can select DMA request line 3, 5, 6, or 7 as shown in Table IV. This will route the DMA request (DRQx) and acknowledge (-DACKx) signals to and from the appropriate DMA controller channel.

TABLE IV. DMA Channel Selection

CR 3, Bits <1,0>	PLX Reference	DMA Channel
00	0	3
01	1	5
10	2	6
11	3	7

An example PC and adapter card configuration is shown in Figure 8. Controller 1 is cascaded with Controller 2. Hence, channel 4 of Controller 2 must be programmed for cascade mode so that whenever Controller 1 requests the bus, Controller 2 will arbitrate for it without executing DMA memory or I/O cycles. The software driver must also program the DMA channel used by the SONIC-16 for cascade mode. This is done by writing the commands shown in Table V to the registers of the appropriate DMA Controller. These commands define the sense of the DROx/DACKx lines, set the arbitration priority algorithm (fixed or rotating), enable and cascade a particular channel and unmask the channel.

TABLE V. DMA Controller Programming

DMA Register	I/O Addr	Data	Channel
COMMAND			
DRQx Active High	08H	10H	3
DACKx Active Low	D0H	10H	5
Rotating Priority	D0H	10H	6
Enable Channel	D0H	10H	7
MODE			
	0BH	D3H	3
	D6H	D1H	5
Cascade Channel	D6H	D2H	6
	D6H	D3H	7
Write Single	0AH	03H	3
Mask Bit	D4H	01H	5
	D4H	02H	6
Unmask Channel	D4H	03H	7

MASTER CYCLES

The following section presents a basic description of the AT bus, AT9010 and SONIC-16 signals generated when the DP83916EB-AT becomes a bus master. It illustrates a timing diagram and presents a basic description of relevant



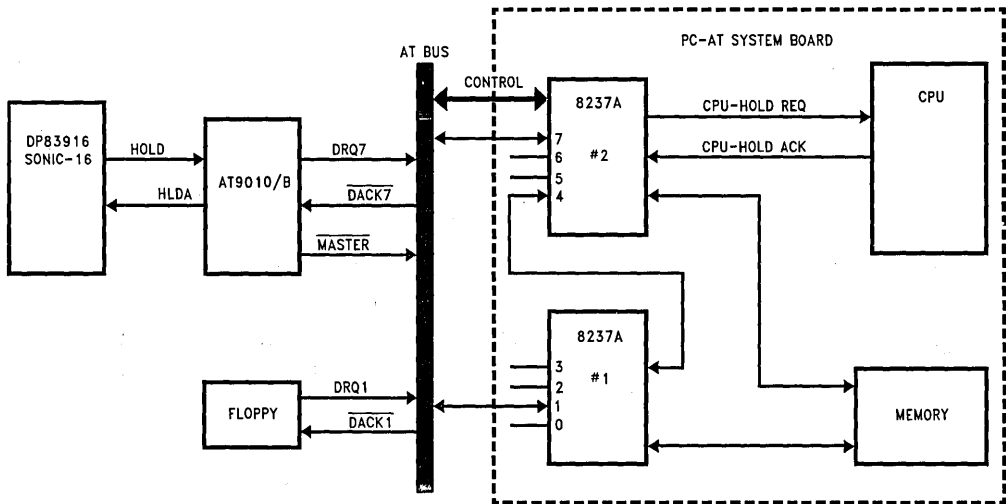


FIGURE 8. Example DMA Controller Configuration

TL/F/11707-6

signals associated with the SONIC-16's read and write cycles to system memory. A more detailed description of the signals associated with master cycles is located in Appendix IV at the end of this application note.

As the DP83916EB-AT transmits and receives packets, it accesses system memory to read and write packet data or descriptor information. The SONIC-16 requires the bus for these operations; hence, it drives HOLD to the AT9010 which translates this request for the bus into an active DRQx line. Once the AT bus is available, the DMA controller responds to the AT9010 via the chosen  $\overline{\text{DACK}}_x$  line. Subsequently, the AT9010 asserts MASTER to the AT bus and relays the  $\overline{\text{DACK}}_x$  to the SONIC-16 by asserting HLDA. This handshake is shown in *Figure 8*.

Once the SONIC-16 has gained ownership of the AT bus, it generates multiple read or write cycles to the AT's system memory. To begin the cycle, the SONIC-16 drives MW-R to the AT9010. Then, the AT9010 drives HAOE to enable the address buffers, HDOE0/HDOE1 to enable the data buffers and HDDIR high (memory write) or low (memory read) to establish direction for the flow of data through the data buffers.

At the beginning of each transfer in the cycle, the SONIC-16 drives an address into the address buffers, asserts address

strobe,  $\overline{\text{ADS}}$ , to the AT9010 and latches the address onto the bus. The rest of the transfer depends on whether the SONIC-16 is writing data or reading data from system memory.

If the SONIC-16 is writing data, the AT9010 will strobe  $\overline{\text{MEMW}}$  low to the bus for each transfer. The SONIC-16 will source data which is valid after the falling edge of the first BSCK of each cycle. Once the system asserts IOCHRDY high, the AT9010 will drive  $\overline{\text{RDY}}_i$  to the SONIC-16 to complete the write cycle. A timing diagram for a master write cycle is shown in *Figure 9*.

If the SONIC-16 is reading data, the AT9010 will pulse  $\overline{\text{MEMR}}$  to the bus for each read transfer. The system memory will source the data which is latched to the SONIC-16 on the rising edge of the first BSCK after  $\overline{\text{RDY}}_i$  is asserted. Again, when the CPU asserts IOCHRDY high, the AT9010 will drive  $\overline{\text{RDY}}_i$  to the SONIC-16 to complete the read cycle.

#### DP83916EB-AT BUS CYCLE CONFIGURATION

There are two timers which govern the SONIC-16's activity on the bus during master mode. The first is a bus hold timer which is activated to ensure the SONIC-16 cannot hog the bus. It begins when the AT9010 receives  $\overline{\text{DACK}}_x$  from the bus and expires after 6 (or 12)  $\mu\text{s}$  depending on whether AT9010 Configuration Register 0, bit 5 is 0 (or 1).



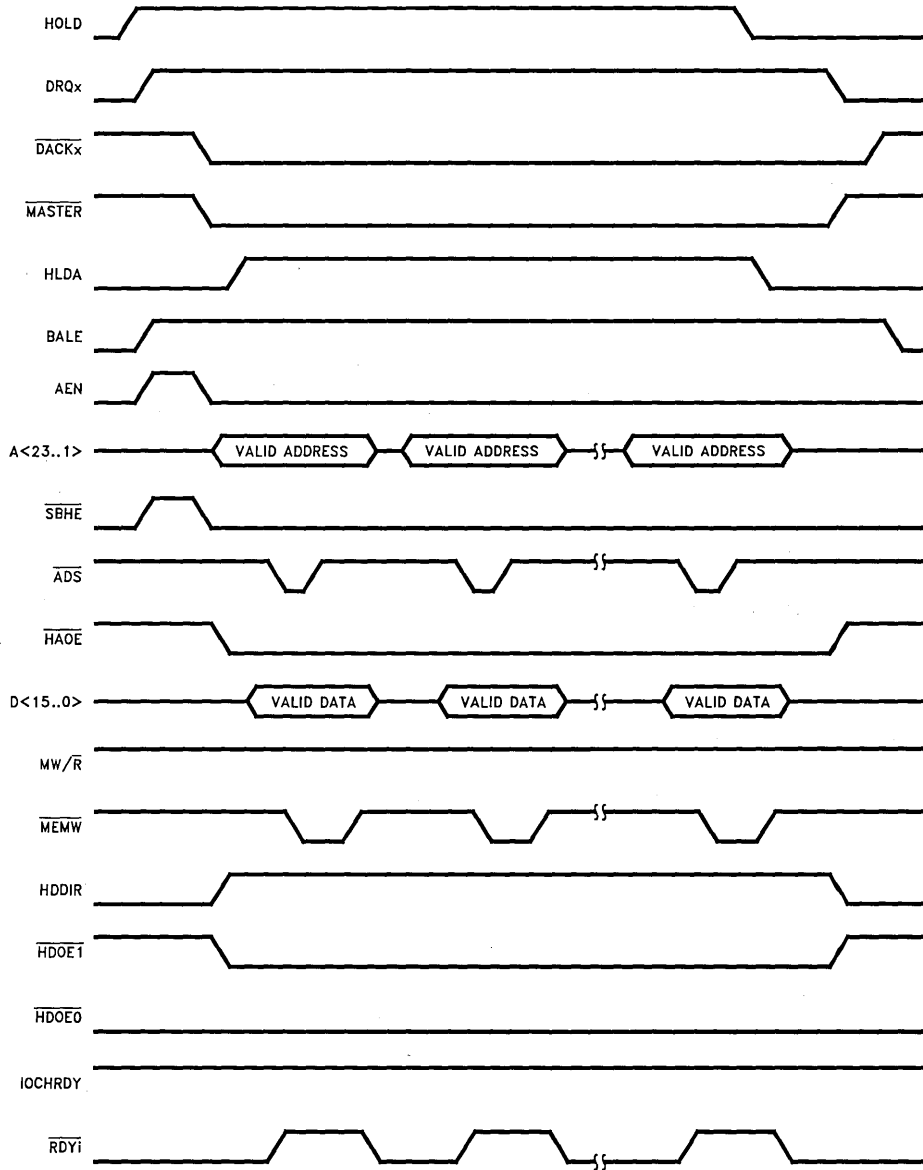


FIGURE 9. Master Write Cycle

TL/F/11707-7

At this point, the AT9010 pre-empts the SONIC off the bus by deasserting HLDA. The original AT9010 chip does not follow proper timing requirements for pre-empting the SONIC-16 and this has been corrected in both the GAL (U11) and the AT9010B. Now, in certain PCs with slow memory cycles, pre-emption causes FIFO underruns or overruns during loopback diagnostics due to the SONIC-16's heavy use of the bus during this mode. Because of these compatibility issues, pre-emption can be disabled. To enable (or disable) the AT9010, set Configuration Register 1, bit 1 to 1 (or 0). This drives a signal PRE-EMPT high (or

low) to the GAL (U11). To enable (or disable) pre-emption in the AT9010B, program CR 1, bit 4 to 0 (or 1).

The second timer is an 800 ns timer. Its purpose is to shorten the SONIC-16's bus acquisition time when the SONIC-16 has control of the bus then deasserts HOLD and re-asserts HOLD before 800 ns has expired. In the original AT9010, if these conditions are satisfied and the 6 (or 12)  $\mu$ s timer has not expired, the original AT9010 will maintain DRQx to the bus. However, if the conditions are satisfied and the 6 (or 12)  $\mu$ s timer has expired, the SONIC will lose the bus once it

deasserts HOLD. This is true regardless of the state of the signal PRE-EMPT. In the AT9010B, the 800 ns timer is independent of the 6 (or 12)  $\mu$ s timer if the 6 (or 12)  $\mu$ s timer is disabled. The AT9010B's 800 ns timer can be enabled (or disabled) by setting CR0, bit 4 to 1 (or 0). For both versions of the AT9010, if the 6/12  $\mu$ s timer is active and has not expired and the 800 ns timer is active, then if the SONIC-16 deasserts HOLD then re-asserts HOLD before 800 ns has expired, HLDA follows HOLD but DRQx is maintained high to the bus.

The AT9010 offers an option of different master transfer cycle speeds. These speeds define the widths of the MEMR and MEMW pulses for each cycle. They are selected in Configuration Register 3 and are outlined in Table VI.

TABLE VI. Master Transfer Cycle Speeds

CR,3 Bits <5,4>	Megabytes/sec
00	5
01	6.7
10	8
11	10

**HUB MANAGEMENT**

The DP83916EB-AT incorporates an interface to the Inter-RIC/Management (IR/M) bus of the DP83950EB-AT. This interface implements a managed hub by connecting the

DP83916EB-AT to one or several DP83950EB-ATs. In this configuration, the SONIC-16 gathers and buffers network statistics for packets sent from the RICs on the DP83950EB-ATs. The SONIC-16 also transmits over the IR/M bus, allowing management statistics to be accessed by any node on the network. A block diagram of the adapter card portion of the managed hub is shown in Figure 10. A comprehensive list of the hub management signals is presented in Appendix IV at the end of this note.

**IR/M RECEIVED PACKET FORMAT**

Packets are sent by RICs on DP83950EB-ATs to the SONIC-16 over the Management portion of the IR/M bus. The format of these packets differs from the format of standard Ethernet packets. First, the preamble and start of frame delimiter of the packets is a 5-bit sequence 01011 rather than the standard eight bytes of 10101....1011.

Second, the packets have Non Return to Zero (NRZ) format because they are sent over the IR/M bus, rather than through the physical layer.

Third, seven bytes of status information are appended by the RICs after the cyclical redundancy check (CRC) sequence of the packet. This information contains statistics regarding the packet's transmission over the network. Because the status field is appended to the end of the packet, the SONIC-16 interprets the last four bytes of status as CRC and flags a CRC error even though there is no legitimate CRC error. For this reason, the SONIC-16's Receive Control Register (RCR) must be programmed to accept packets with errors.

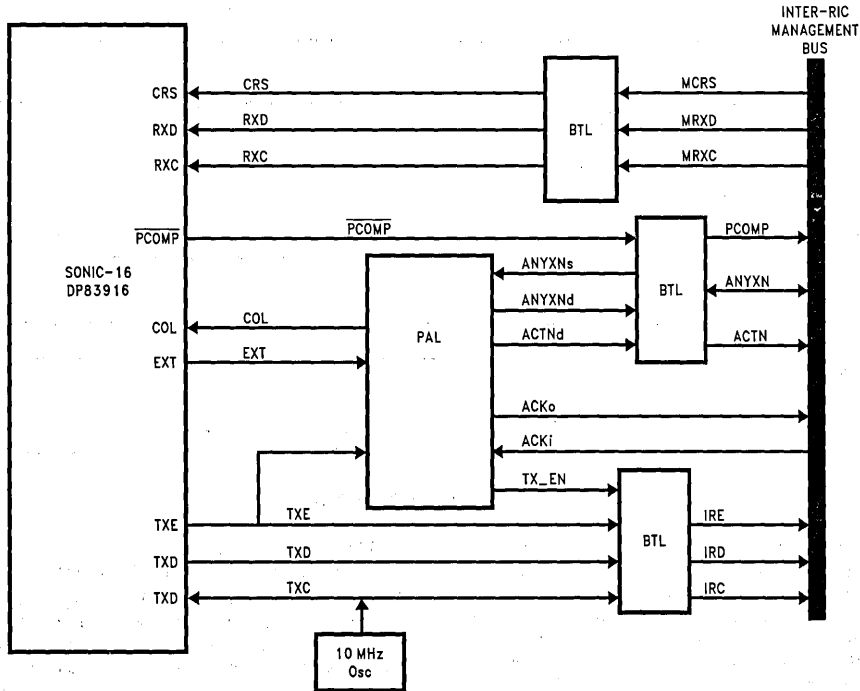


FIGURE 10. Hub Management Interface

TL/F/11707-8

Finally, the packet's destination may specify the SONIC-16 node if the packet contains management commands intended for the SONIC-16 or another node if the packet is sent to the SONIC-16 for the purpose of saving the status information. Hence, the SONIC-16's RCR must be configured to accept all packets including runt packets and all address type packets (in addition to accepting errored packets as described above).

A packet that is sent over the management bus has the format shown in Figure 11. A detailed description of the information contained in the seven bytes of management statistics is located in the DP83950 Repeater Interface Controller (RIC) Data Sheet.

**PACKET RECEPTION SIGNALS**

The signals MCRS, MRXD and MRXC sent over the management bus, are the management carrier sense, management data, and management clock which specify the packets sent to the SONIC-16. They are buffered through inverting turbo transceivers (BTLs). The BTLs drive the IR/M bus signals with the same characteristics as the signals on the DP83950EB-AT card. BTL features include high density backplane capabilities, minimum delay and fast voltage switching characteristics; however, BTLs are not required for all applications.

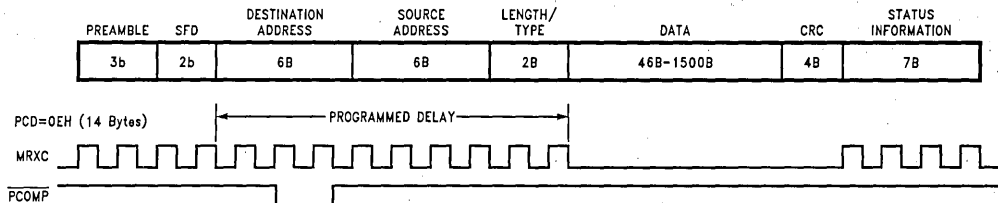
As described above, the packets are in NRZ format; hence, the buffered signals, CRS, RXD and RXC driven to the SONIC-16 are NRZ signals. These signals are connected directly to the CRS, RXD and RXC of the SONIC-16. The SONIC-16 is programmed (USER\_PIN3 of AT9010 Configuration Register 1 is set to 1) for external ENDEC mode.

With the above configuration, the SONIC-16 will buffer every packet received from the IR/M bus. It can, however, be programmed to save memory space by "compressing" packets whose data is not intended for reception by the SONIC-16. With this feature, the SONIC-16 can buffer status information, a portion of data and status or the entire packet. This is accomplished via a SONIC-16 control signal called PCOMP. If PCOMP is asserted to the management bus, the receive clock signal, MRXC, will be inhibited during transmission of packet data and enabled during transmission of the packet's seven bytes of status information. This causes the packet to be "compressed"; i.e., only destination address and status data are buffered to memory.

In order to use packet compression, the SONIC-16's DCR2 register must be programmed to assert PCOMP upon CAM (Content Addressable Memory) register match or mismatch. For a managed hub, the DCR2 must be programmed to assert PCOMP upon CAM mismatch. Then, if the SONIC-16's CAM is programmed with its own Ethernet address, all packets with destination addresses equal to the SONIC-16's address will be buffered. All other packets will be compressed. For a managed bridge, the DCR2 register must be programmed to assert PCOMP upon CAM register match. In this case, the SONIC-16's CAM is programmed with the addresses of all RICs on the LAN from which the packet is being transmitted. Then, only packets intended for the SONIC-16 or for nodes on the other side of the bridge link will be buffered. All packets which are merely repeated to the RIC connected to the SONIC-16 then forwarded to the SONIC-16 over the Management bus, will be compressed to save status information only.

In addition to programming the SONIC-16's DCR2 the Packet Compress Decode (PCD) Register of the RIC must be initialized with the number of bytes after SFD, not including seven bytes of status information, to be transferred if the SONIC-16 asserts PCOMP. According to the DP83950 RIC Data Sheet, the value of this register must be less than or equal to 255 bytes. The actual value, however, must be between 7-255 bytes because the SONIC-16 requires six bytes of destination address and five bits of address compare time in order to determine whether or not to assert PCOMP. If the user enters a PCD value less than seven bytes, the driver software should change the value to seven so that PCOMP will operate properly. In this scenario, six bytes of destination address and one byte of source address will be buffered along with the seven bytes of status information.

An example of PCOMP and the effect on MRXC is shown in Figure 11. For this example, the RIC's PCD is 0EH. As it transmits the packet, the RIC counts 14 bytes from the beginning of the destination address. Because the SONIC-16 asserts PCOMP, the RIC inhibits the MRXC 14 bytes after the beginning of the packet. It enables the MRXC for the last seven bytes of status data.



**FIGURE 11. Packet Reception Over Management Bus**

TL/F/11707-9

## IR/M PACKET RECEPTION

When the SONIC-16 begins receiving a packet, it compares the destination address of the packet with all addresses in its content addressable memory (CAM) register. If the SONIC-16 is configured as a managed hub and there is a CAM match, the entire packet, along with the seven bytes of status information, is buffered to memory where it waits to be processed by upper-level management software. If there is no CAM match, the SONIC-16 asserts  $\overline{PCOMP}$  to the Management bus. Depending on the value of the PCD, the RIC sending the packet will inhibit the receive clock signal, MRXC during a portion of the packet data and enable it during the seven bytes of status. Hence, only destination address, a portion of source address/data and status information is buffered to memory where it waits to be processed by upper-level management software.

## IR/M TRANSMITTED PACKET FORMAT

Packets are sent by the SONIC-16 to the RICs on DP83950EB-ATs over the Inter-RIC portion of the IR/M bus. These packets have the format of standard Ethernet packets.

## PACKET TRANSMISSION SIGNALS

The SONIC-16 transmits over the Inter-RIC bus using the transmit enable (TXE), transmit data (TXD) and transmit clock (TXC) to specify the packets. TXC is driven by a 10 MHz signal from a flip-flop which divides an external 20 MHz oscillator clock by two. The transmit signals are connected to the SONIC-16's transmit pins, TXE, TXD and TXC and have NRZ format. They are driven through an inverting turbo transceiver (BTL) and become the Inter-RIC enable (IRE), Inter-RIC data (IRD) and Inter-RIC clock (IRC), which connect directly to the Inter-RIC bus.

Because of the SONIC-16's interface to the Inter-RIC/Management bus, it appears to be another RIC to the rest of the RIC network. Hence, the SONIC-16 participates in the RICs' serial arbitration scheme for transmission and uses the same handshake signals. This arbitration scheme is contained in a PAL and is described in the following paragraphs. The actual PAL equations are located at the end of this application note.

The RICs and SONIC-16 are connected in the serial arbitration scheme by two signals, ACKi and ACKo. ACKo of a RIC above the SONIC-16 connects to ACKi of the SONIC-16; ACKo of the SONIC-16 connects to ACKi of a RIC below it. The SONIC-16 and RICs pass permission to transmit down the chain by driving ACKo high to the ACKi of the next chip in the chain.

Regardless of whether or not the SONIC-16 has permission to transmit, it does so whenever the management bus is quiet and there is data to send. Hence, when the SONIC-16 wants to transmit, it drives TXE high. If the SONIC-16 has permission to transmit (i.e., ACKi is high), the PAL activates ACTNd high to notify the RICs of the SONIC-16's transmit activity on the Inter-RIC bus. If the SONIC-16 does not have permission to transmit (i.e., ACKi is low), the PAL activates ANYXNd high which notifies the RICs of a SONIC-16 transmit collision.

The SONIC-16's collision pin (COL) is driven by the PAL when the SONIC-16 is transmitting and either a transmit collision occurs on the network/Inter-RIC bus or a receive collision occurs on a RIC's AUI port. If the SONIC-16 is transmitting and a collision occurs, the SONIC-16 sends jam pattern, then backs off the Inter-RIC bus. At the same time, the RICs send jam pattern and then become idle. After some time, the SONIC-16 attempts to retransmit. If the SONIC-16 is not transmitting and a collision occurs on the network, the RICs send jam to their ports. A Management Interface Configuration (MIFCON) bit in the RIC's Interrupt and Management Configuration Register determines the outcome of this collision event. If MIFCON is 0 and the collision occurs before the packet's start of frame delimiter, the RIC whose packet has collided, will send 01011 followed by seven bytes of status (which reflect the collision) to the SONIC-16. If MIFCON is 1 and the collision occurs before the SFD, neither packet nor status data is transmitted over the Management bus to the SONIC-16. Finally, if MIFCON is 0 (or 1) and the collision occurs after the packet's SFD, the RIC appends the status information and sends the packet to the SONIC-16.

The PAL drives a BTL transmit enable (TX\_EN) signal to the transmit BTL which enables the BTL only when the SONIC-16 is configured for an external ENDEC, has permission to transmit and wants to transmit. This prevents the BTL from driving IRE, IRD and IRC unless the SONIC-16's transmission is valid.

## DP83916EB-AT CONFIGURATION FOR HUB MANAGEMENT

The adapter card must be configured differently to use the hub-management option. First, the card cannot be connected to Inter-RIC/Management bus and a physical layer interface at the same time. Jumper 4 must be disconnected. Second, the AT9010's USER\_PIN3 (bit 3) in Configuration Register 1 must be set to a 1 to configure the SONIC-16 for external ENDEC mode. Third, the AT9010's USERPIN2 (bit 2) in Configuration Register 1 must be set to a 0 to enable the receiving BTL.

## MEDIA INTERFACE

The network interface of the DP8392C Coaxial Transceiver Interface (CTI) as a coaxial cable line driver/receiver connected between the SONIC-16 and the BNC connector for Thin Ethernet coaxial cable. For transmission, it converts AUI signals to single-ended 10BASE2 signals. On reception, it converts single-ended 10BASE2 signals to AUI signals. The isolation between the CTI and the SONIC-16, required by the IEEE 802.3, is satisfied on the signal lines by a transformer. Power isolation for the CTI is performed by a DC to DC converter which supplies the CTI with a -9V power supply for operation. To use the adapter card in a Thin Ethernet environment, it is necessary to short JP4 which supplies the CTI with -9V.

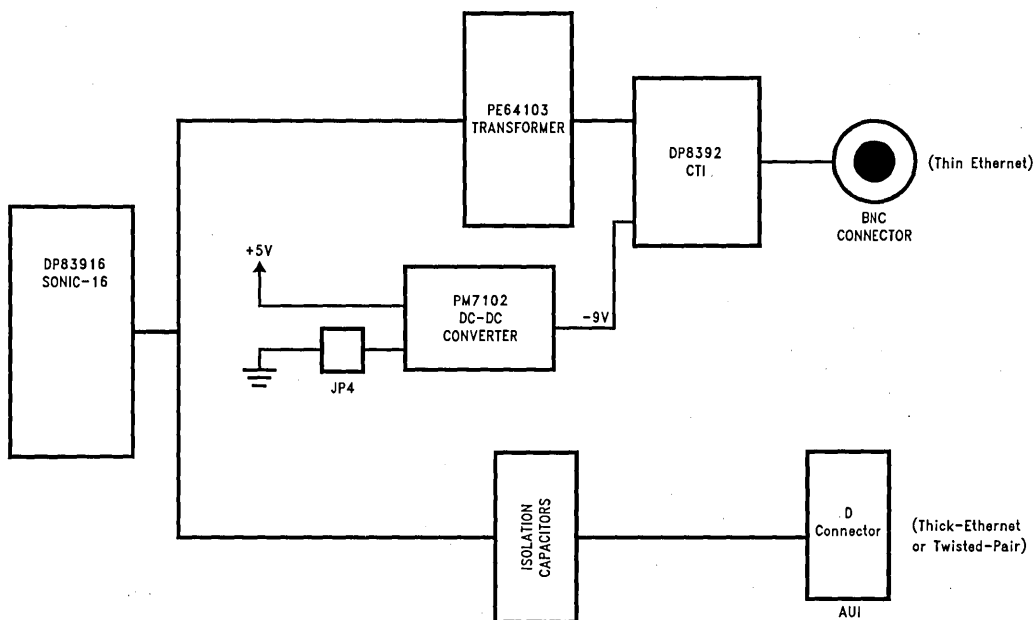


FIGURE 12. Media Interface Block Diagram

TL/F/11707-10

The AUI Interface option allows the use of the DP83916EB-AT with one of several alternative cable media. Possible choices include Thick Ethernet cable for 10BASE2 networks and Twisted-Pair cable for 10BASE-T networks. No on-board transformers are required for isolation because the connector to the AUI is a medium attachment unit (MAU) which houses its own transformer and DC to DC converter. However, capacitors are used for DC isolation and 16V fault tolerance. To use the AUI interface, open JP4 to disable power to the CTI.

The interface options and jumper settings are summarized in Table VII. It is imperative to note that **only one network interface option and cable can be used at a time. Multiple cables will result in network errors.**

TABLE VII. Jumper Selection for Media Interface or Hub Management

JP4	Network Interface/HM
Short*	Thin Ethernet*
Open	Thick Ethernet, Twisted-Pair via AUI
Open	HUB-Management

\* DP83916EB-AT default setting

#### DP83916EB-AT CONFIGURATION

In order to maximize the utility and options of the DP83916EB-AT adapter card, it is imperative that the board is configured correctly. The following section highlights the **hardware and software configuration issues which must be considered prior to installation of the card for the first time.**

#### HARDWARE CONFIGURATION

There are two versions of the PLX bus interface chip: the AT9010 and the AT9010B. The DP83916EB-AT design supports both chips with the following component placement considerations. If the original AT9010 is used, a GAL (U11) containing AT9010 fixes must be populated. In addition, resistors R46, R47 and R48 must not be populated. If the AT9010B is used, the GAL (U11) must be left open and resistors R46, R47 and R48 must be populated. The above considerations are determined at the time of board assembly. Hence, they should not be of concern to the card user.

The user must determine the I/O address of the card and set jumpers JP1-JP3. If the optional boot EPROM is to be used, it must be inserted on the adapter card.

To use the DP83916EB-AT's media interface, the user must select one of the interface options including Thin Coax (10BASE2) or AUI (with 10 BASE5 or 10BASE-T) and configure JP4 according to the Media Interface Section. Or, to use hub management, the user must install a DP83950E-B-AT RICKIT, disconnect any DP83916EB-AT media interface connections and open JP4.

The above hardware settings must be considered prior to inserting the board into an AT bus slot.

#### SOFTWARE CONFIGURATION

The DP83916EB-AT features many user options which can be selected by programming the configuration registers of the AT9010. The following summarizes the software options available; references to AT9010 are given as (Configuration Register Number, Applicable Bits). For a complete listing of AT9010 Configuration Registers and demonstration software defaults, refer to Appendix III.

**Card enable** (CR0, Bit0): enables (1) or disables (0) the original AT9010. Once disabled, the original AT9010 can only be re-enabled with hard reset. The AT9010B is always enabled regardless of the state of this bit.

**Selectable interrupt lines** (CR0, Bits <2,1>): one of four must be chosen: 00 = IRQ3, 01 = IRQ4, 10 = IRQ5, 11 = IRQ9.

**Interrupt unmask** (CR0, Bit 3): unmask (1) or masks (0) the IRQx signal when INT is driven by the SONIC-16.

**800 ns timer** (CR0, Bit 4): enables (1) or disables (0) the 800 ns timer in AT9010B. This timer is always enabled in the original AT9010.

Maximum bus hold time after detection of  $\overline{\text{DACKx}}$  (CR0, Bit 5): 0 = 6  $\mu\text{s}$ , 1 = 12  $\mu\text{s}$ . In the original AT9010, this timer can be enabled (or disabled) by setting CR1, bit 1 to 1 (or 0). In the AT9010B, it can be enabled (or disabled) by setting bit 4 of CR1 to 0 (or 1).

**USER\_PIN1/PRE-EMPT** (CR1, Bit 1): enables (1) or disables (0) SONIC-16 pre-emption in the original AT9010.

**USER\_PIN2/RE\_EN** (CR1, Bit2): drives a 0 for a BTL receive enable in the hub management interface. This bit must be a 1 when not using hub management.

**USER\_PIN3/EXT** (CR1, Bit3): drives a 0 for an internal SONIC-16 ENDEC when using the physical interface or a 1 for an external ENDEC when using the hub management interface.

**I/O address of the card** (CR1, Bits <7..5>): one of seven choices must be made; these are outlined in the Slave Logic Section. Note, the address programmed in the AT9010's CR1 must match the address selected by JP1-JP3.

**EPROM memory size or disable** (CR2, Bits <7,6>): selects a 16k EPROM, 8k EPROM or EPROM disable: 01 = 16k, 10 = 8k, 11 = disable.

**EPROM memory address** (CR2, Bits <5..2>): this address must be specified if using the EPROM. Details are given in the Slave Logic Section.

**Selectable DMA lines** (CR3, Bits <1,0>): one of four must be chosen: 00 = DMA 3, 01 = DMA 5, 10 = DMA 6 11 = DMA7.

**MEMW cycle extension** (CR3, Bit 2): when set to 0, this option will extend the -MEMW cycle by 50 ns to allow additional address set-up time.

**IOCHRDY assert** (CR3, Bit 3): selects IOCHRDY normal assert (0) or early assert (1) in AT9010B only. See Slave Logic Section for details.

**Master data transfer cycle** (CR3, Bits <5,4>): one of four speeds must be chosen: 00 = 5, 01 = 6.7, 10 = 8, 11 = 10 MB/sec.

**Channel check enable:** (CR3, Bit 6) is not used by the DP83916EB-AT. This bit should be set to 1 to disable the AT9010 check output.

**Software reset** (CR15, Bit 7): is supported in the AT9010B. When this bit is set to 1, the AT9010B resets all of the chip's functions to their default conditions except CR0-3 and CR15, bits 3-5. This bit is cleared by writing a 0 to CR15, bit 7 to clear the bit. In the original AT9010, setting this bit to 1 causes the card to be lost in I/O space; it can only be recovered by hard reset.

#### ADDITIONAL INFORMATION

Additional information for the AT9010 Bus Master Interface Chip is available through PLX Technology, Inc., 625 Clyde Avenue, Mountain View, CA 94043, (415) 960-0448, FAX (415) 960-0479.

**APPENDIX I: PAL (U16) AND GAL (U11) EQUATIONS**

This appendix provides the equations for the PAL (U16) and the GAL (U11). The PAL contains arbitration equations for hub management. The GAL contains fixes for the bugs in the original AT9010 chip.

**DP83916EB-AT Inter-RIC/Management Interface PAL (U16)**

```
module intf2
```

```
title 'SONIC/RIC Management Interface for DP83916EB-AT (r2)'
```

```
"history: finalized by Bill Bunch on 9/19/91
```

```
"device declaration
```

```
    U16_PAL2 device 'p16l8';
```

```
"inputs
```

```
    anyxn_s  pin 2;
    ack      pin 3;
    ext      pin 4;
    txe      pin 9;
    coln     pin 17;
```

```
"outputs
```

```
    col      pin 12;
    tx_en    pin 13;
    acko     pin 14;
    actn_d   pin 18;
    anyxn_d  pin 19;
```

```
equations
```

```
    acko = !txe & acki;
```

```
    actn_d = txe & acki;
```

```
    anyxn_d = txe & !acki;
```

```
    col = txe & (anyxn_s # coln);
```

```
    tx_en = txe & acki & ext;
```

```
    col.oe = ext;
```

```
    actn_d.oe = ext;
```

```
    anyxn_d.oe = ext;
```

```
end intf2;
```

TLJ/F/11707-11

## DP83916EB-AT Original AT9010 GAL Fixes (U11)

```
module splxr2
```

```
flag '-r1','-t4';
```

```
title 'GAL fixes for original PLX AT9010 bugs and DP83916EB-AT - p/n U11'
```

```
"history: Finalized by Denise Troutman on 6/1/92
```

```
"This GAL fixes the following problems in the original AT9010 chip:
```

- ```
" 1. Asserting IOCHRDIAT inappropriately when certain PCs drive the SONIC-16's I/O address
" during boot-up
" 2. Not maintaining HLDA long enough to satisfy the SONIC-16's pre-emption specification
" in the following case: the SONIC-16 requests the bus (drives HOLD high) within 800 ns of
" giving up the bus (driving HOLD low) and the 6/12 μs timer has just expired
" 3. LRESET with inverted polarity
" 4. Inability to disable SONIC-16 pre-emption off the bus - this causes problems in certain PCs
" with slow memory whereby the memory transfers are so long that FIFO underruns during
" transmission or FIFO overruns during reception may occur; the problem is aggravated during
" loopback because of the SONIC-16's heavy use of the bus for this mode.
```

```
"device declaration
```

```
U11_GAL2 device 'P20V8R';
```

```
@page
```

```
"pin assignments
```

```
"INPUTS
```

```
aen          pin 8;
liord        pin 2;
liowr       pin 3;
!hostown    pin 4;
lclk        pin 5;
hold        pin 6;
hlda       pin 7;
preempt    pin 9;
lreset     pin 10;
```

```
"outputs
```

```
hldadlym    pin 16;
hldadly     pin 17;
hldaout     pin 22;
!iochrdoe   pin 19;
!iochrdyat  pin 20;
!iochrdybus pin 21;
hldasonic   pin 18;
reset       pin 15;
```

```
iochrdoe    istype 'com,neg';
iochrdyat   istype 'com,neg';
iochrdybus  istype 'com,neg';
hldadlym    istype 'com,neg';
hldadly     istype 'com,neg';
hldaout     istype 'com,neg';
hldasonic   istype 'com,neg';
reset       istype 'com,neg';
```



"constant declarations

```
L = 0;
H = 1;
OFF = 0;
ON = 1;
X = .X.;
Z = .Z.;
CK = .C.;
```

@page  
equations

"Qualify IOCHRDIAT with  $\overline{IORD}$  and  $\overline{IOWR}$   
iochrdyoe = !hostown & !aen & iochrdyat & (iord # iowr);

```
iochrdybus.OE = iochrdyoe;
iochrdybus = ON;
```

```
iochrdyat.OE = hostown;
iochrdyat = iochrdybus;
```

"Delay HLDA from AT9010 up to one clock  
hldadlym = (!lclk & hlda) # (lclk & hldadlym) # (hlda & hldadlym);

```
hldadly = (lclk & hldadlym) # (!lclk & hldadly) # (hldadlym & hldadly);
```

```
hldaout = hlda # (hold & hldadly);
```

"Enable or disable pre-emption based on state of PREEMPT  
hldasonic = (preempt & hldaout) #  
(!preempt & (hlda # hldadly # (hold & hldasonic)));

"Invert RESET to SONIC-16  
reset = !reset;

end splxr2;

TL/F/11707-13

**APPENDIX II: BILL OF MATERIALS (BOM) for DP83916EB-AT**

This appendix provides a list of all components placed on the DP83916EB-AT. If the original AT9010 is populated, a GAL (U11) must also be populated and resistors R46, R47 and R48 should be left open. If the AT9010B is populated, the GAL (U11) should be left open and R46, R47 and R48 should be populated.

**Capacitors (58)**

|                      |                   |                  |
|----------------------|-------------------|------------------|
| C3..C6               | 0.1 $\mu$ F/50V   | 10% Monolithic   |
| C7                   | 0.01 $\mu$ F/25V  | 20% Monolithic   |
| C8                   | 0.01 $\mu$ F/1 kV | 10% Ceramic Disk |
| C9                   | 0.75 pF/1 kV      | Spark Gap        |
| C10, 11              | 0.01 $\mu$ F/50V  | 10% Monolithic   |
| C12                  | 47 $\mu$ F/50V    | 20% Tantalum     |
| C13. .C25, C28. .C39 | 0.1 $\mu$ F/50V   | 20% Monolithic   |
| C26, C27, C40. .C48  | 4.7 $\mu$ F/16V   | 20% Tantalum     |
| C49                  | 4.7 $\mu$ F/25V   | 20% Tantalum     |
| C50. .C54            | 0.1 $\mu$ F/50V   | 20% Tantalum     |
| C55 .C60             | 1 $\mu$ F/50V     | 10% Monolithic   |

**Resistors (46) (5%, 1/4W unless otherwise specified)**

|                     |      |                                                                |
|---------------------|------|----------------------------------------------------------------|
| R1. .R4, R9, R12    |      |                                                                |
| R16, R17, R30. .R45 | 4.7k |                                                                |
| R5. .R8             | 10k  |                                                                |
| R10                 | 1k   |                                                                |
| R18, R20. .R23      | 1k   | 1%, 1/4W                                                       |
| R14, R15            | 270  |                                                                |
| R19                 | 150  | 1%, 1/4W                                                       |
| R24                 | 10k  | 1%, 1/4W                                                       |
| R25                 | 1M   | 5%, 1/2W                                                       |
| R26. .R29           | 39.2 | 1%, 1/4W                                                       |
| R46. .R48           | 0    | (do not populate for original AT9010;<br>populate for AT9010B) |

**Integrated Circuits (17)**

|               |             |                                                                |
|---------------|-------------|----------------------------------------------------------------|
| U1, U2        | DM74AS245   |                                                                |
| U3            | DM74S288    |                                                                |
| U4            | NMC27CP128  | (not supplied on board)                                        |
| U5            | DM74ALS244A |                                                                |
| U6. .U8       | DM74ALS541  |                                                                |
| U9            | DM74ALS521  |                                                                |
| U10           | PLXAT9010/B |                                                                |
| U11           | GAL20V8A-15 | (populate for original AT9010; do not<br>populate for AT9010B) |
| U12           | DP83916B    |                                                                |
| U14           | 74F74       |                                                                |
| U15, U17, U18 | DS3893A BTL |                                                                |
| U16           | PAL16L8A    |                                                                |
| U19           | DP8392V     |                                                                |

**Connectors (2)**

|    |                            |                        |
|----|----------------------------|------------------------|
| J4 | BNC Connector, socket      | (AMP #227161-2)        |
| J5 | 15-Pin D Connector, socket | (AMP 9020A # 747845-4) |

**Magnetics (2)**

|     |                                               |
|-----|-----------------------------------------------|
| T1  | PE64103 (Pulse Engineering) or LT6003 (Valor) |
| U20 | PM7102 (Valor) DC-DC Converter                |

**Jumpers (4)**

|          |                                                    |
|----------|----------------------------------------------------|
| JPI..JP4 | Single Jumpers 1 x 2 Shunt Block with 0.1" spacing |
|----------|----------------------------------------------------|

**Test Pins (35)**

|           |                  |
|-----------|------------------|
| TP1..TP35 | Single post pins |
|-----------|------------------|

**Sockets (5)**

|    |                                               |                 |
|----|-----------------------------------------------|-----------------|
| S1 | 24-pin, Dual in-line socket for GAL (U11)     |                 |
| S2 | 20-pin, Dual in-line socket for PAL (U16)     |                 |
| S3 | 16-pin, Dual in-line socket for the PROM (U3) |                 |
| S4 | 28-pin, Dual in-line socket for EPROM (U4)    |                 |
| S5 | 132-pin AMP Socket for SONIC-16 (U12)         |                 |
|    | Housing Sub-Assembly Cover                    | (AMP #821949-5) |
|    | Cover                                         | (AMP #821942-1) |

**Others (6)**

|                 |                                                  |                  |
|-----------------|--------------------------------------------------|------------------|
| U13             | 20 MHz Oscillator 40%–60% Duty, 0.001% Tolerance |                  |
| D1              | MMBD1203 Diode                                   |                  |
| Bracket         | Face plate                                       |                  |
| Slide Latch Kit | For 15-pin D-Connector (J5)                      | (AMP # 745583-5) |

Note: U4 (EPROM) is marked "not supplied on board"; the component socket is left open.

**APPENDIX III: AT9010 and AT9010B CONFIGURATION REGISTERS**

This appendix describes the features and programming for the original AT9010 and AT9010B Configuration Registers. It can be assumed that the bits have the same function in both chips unless otherwise noted. The demonstration software "sonicpla.exe" defaults are also provided.

**CONFIGURATION REGISTER 0: default = 71 H**

| Bit        | 7   | 6   | 5   | 4      | 3  | 2  | 1  | 0      |
|------------|-----|-----|-----|--------|----|----|----|--------|
| Short Name | RES | RES | BHT | RES/ET | UI | IS | IS | CE/RES |
| Default    | 0   | 1   | 1   | 1      | 0  | 0  | 0  | 1      |

RES = reserved

BHT = bus hold time (0 = 6  $\mu$ s/1 = 12  $\mu$ s)

RES/ET = AT9010: reserved

AT9010B: enable 800 ns timer (1 = enable/0 = disable)

UI = unmask interrupt (1 = unmask/0 = mask)

IS = interrupt select (00 = IRQ3, 01 = IRQ4, 10 = IRQ5, 11 = IRQ9)

CE/RES = AT9010: card enable

AT9010B: reserved

**CONFIGURATION REGISTER 1: default = 94 H**

| Bit        | 7   | 6   | 5   | 4      | 3   | 2     | 1  | 0  |
|------------|-----|-----|-----|--------|-----|-------|----|----|
| Short Name | IOA | IOA | IOA | RES/DT | EXT | RE_EN | PE | U0 |
| Default    | 1   | 0   | 0   | 1      | 0   | 1     | 0  | 0  |

IOA = I/O base address (000 = 100 H, 001 = 120 H, 010 = 140 H, 011 = 160 H,  
100 = 300 H, 101 = 320 H, 110 = 340 H, 111 = 340 H)

RES/DT = AT9010: reserved

AT9010B: disable 6/12  $\mu$ s bus hold timer (1 = disable/0 = enable)

EXT = external ENDEC for Inter-RIC/Management (0 = internal/1 = external)

RE\_EN = enable receive for Inter-RIC/Management (0 = enable 1 = disable)

PE = pre-empt enable (1 = enable/0 = disable SONIC pre-emption; AT9010)

U0 = User bit 0 (not used)

**CONFIGURATION REGISTER 2: default = C0 H**

| Bit        | 7  | 6  | 5   | 4   | 3   | 2   | 1   | 0   |
|------------|----|----|-----|-----|-----|-----|-----|-----|
| Short Name | PS | PS | PBA | PBA | PBA | PBA | RES | RES |
| Default    | 1  | 1  | 0   | 0   | 0   | 0   | 0   | 0   |

PS = PROM select - size or disable (00 = 32k, 01 = 16k, 10 = 8k, 11 = disable)

PBA = PROM base address (for 16k EPROM: 000X = C0000 H, 001X = C4000 H,  
010X = C8000H, 011X = CC000 H, 100X = D0000 H, 101X = D4000H,  
110X = D8000 H, 111X = DC000 H)

RES = reserved

**CONFIGURATION REGISTER 3: default = C2 H**

| Bit        | 7   | 6   | 5   | 4   | 3      | 2   | 1   | 0   |
|------------|-----|-----|-----|-----|--------|-----|-----|-----|
| Short Name | CCA | MCC | MDT | MDT | RES/IS | ETO | DMA | DMA |
| Default    | 1   | 1   | 0   | 0   | 0      | 0   | 1   | 0   |

CCA = channel check (output) assert (not used)

MCC = mask channel check (not used)

MDT = master data transfer cycle speed (00 = 5, 01 = 6.7, 10 = 8, 11 = 10 MB/sec)

RES/ISA = AT9010: reserved

AT9010B: IOCHRDY signal assert (0 = normal/1 = early IOCHRDY signal)

ETO = extra time off (0 = extended/1 = normal -MEMW cycle)

DMA = DMA channel select (00 = DMA3, 01 = DMA 5, 10 = DMA 6 11 = DMA 7)

**CONFIGURATION REGISTER 15: default = 00 H**

| Bit        | 7      | 6   | 5  | 4  | 3  | 2  | 1   | 0   |
|------------|--------|-----|----|----|----|----|-----|-----|
| Short Name | RES/SR | CSI | P5 | P4 | P3 | II | RES | RES |
| Default    | 0      | 0   | 0  | 0  | 0  | 0  | 0   | 0   |

RES/SR = AT9010: reserved

AT9010B: software reset (1 = soft reset/0 = non-soft reset mode)

CSI = channel check (input) indicator (not used)

P5, P4, P3 = SONIC register page select - bit 5, bit 4, bit 3

II = interrupt indicator

RES = reserved

#### APPENDIX IV: DP83916EB-AT SIGNALS

This appendix presents a detailed description of the adapter card control signals specific to timing, slave cycles and initialization, master cycles and hub management. They are presented in the form SIGNAL (ORIGIN, DESTINATION) or SIGNAL (ORIGIN to DESTINATION/ORIGIN to DESTINATION).

#### CLOCK SIGNALS

The clock signals are provided for the synchronous operations of the AT9010, the SONIC-16 and the optional IR/M interface. They are described in detail below:

**20 MHz** (20 MHz Osc, AT9010 and SONIC-16 and Flip-flop) is an oscillator signal which drives all synchronous operations in the AT9010 and provides a clock for the SONIC-16's ENDEC. It also provides a 20 MHz signal to the flip-flop divide by two circuit which drives 10 MHz to the transmit BTL.

**BSCK** (AT9010, AT bus and GAL) provides timing for the SONIC-16 DMA logic.

**TXC** (Flip-flop, SONIC-16 and Tx BTL) provides the timing for the transmission of packets when using the hub-management interface.

#### SLAVE CYCLE SIGNALS

The following control signals provide interrupt, reset and status functions:

**INT** (SONIC-16, AT9010) is active when the SONIC-16 asserts an interrupt request.

**IRQx** (AT9010, AT bus) is asserted by the AT9010 when the SONIC-16 asserts its interrupt request line, INT. One of four lines IRQ9, IRQ5, IRQ4 or IRQ3 is selected as the interrupt by programming bits <2,1> of the AT9010's Configuration Register 0.

**RES\_DRV** (AT bus, AT9010) provides a hard reset to the AT9010. This signal initializes logic internal to the AT9010.

**LRESET/-RESET** (AT9010 to GAL to SONIC-16) provides a hardware reset to the SONIC-16. It is asserted and deasserted synchronous to BSCK. For the AT9010B, this signal is inverted inside the chip and is driven directly to the SONIC-16.

**S<2..0>** (SONIC-16, AT9010) are SONIC-16 status lines which indicate the current SONIC-16 bus operation.

**USER\_PIN3/EXT** (AT9010, SONIC-16 and PAL) drives the EXT pin input of the SONIC-16 low to enable the SONIC-16's internal ENDEC (when using a media interface) and high to disable the ENDEC (for managed-hub applications). The level of this signal is set in the AT9010's Configuration Register 1.

**USER\_PIN2/-RE\_EN** (AT9010, BTL) drives the receive enable signal RE\_EN, to one of the turbo-transceivers (BTL) when using the SONIC-16/RIC hub-management interface.

**POSCS3** (AT9010, A-buffer) enables an address buffer to load the address of the adapter card into AT9010 Configuration Register 1, bits <7..5> after hard reset. All other POSCS pins are not connected because the registers are loaded by software rather than by hardware.

The following signals are utilized during slave mode for both I/O cycles and EPROM cycles:

**BALE** (AT bus, AT9010) is driven high for all slave cycles on the DP83916EB-AT.

**HDDIR** (AT9010, D-Buffers) is an input to the data buffers which identifies the direction of a data transfer. It drives high for data transfers to the AT bus during an I/O or EPROM read cycle. It drives low for data transfers to the adapter card during an I/O write cycle.

**IOCHRDYAT/IOCHRDYBUS** (AT9010 to GAL to AT bus) are input and output signals for both the AT bus and the AT9010. For the original AT9010, the GAL translates the AT9010's IOCHRDYAT into IOCHRDYBUS and drives this signal to the AT bus to complete a 16-bit I/O cycle to the SONIC-16's registers. For the AT9010B, IOCHRDY is driven from the AT9010B directly to the AT bus. An early IOCHRDY signal can be driven by the AT9010B. This is detailed in the Slave Logic Section.

During I/O cycles, the following signals are generated:

**AEN** (AT bus, AT9010 and GAL) is a signal asserted high to all ports during DMA cycles to prevent I/O resources that do not have an active DACKx from responding to DMA controller I/O cycles. The AT9010 uses the low level of AEN to qualify CPU accesses to the registers in I/O space.

**IORD** (AT bus, AT9010 and GAL) indicates that the system is reading data from an I/O register.

**IOWR** (AT bus, AT9010 and GAL) indicates that the system is writing data to an I/O register.

**CS** (AT9010, SONIC-16) is the chip select to the SONIC-16.

**IOCS16** (AT9010, AT bus) is driven to the AT bus when SONIC-16 registers are to be accessed; it indicates a 16-bit slave device.

**PROMID** (AT9010, PROM) is the chip enable for the PROM.

**SAS** (AT9010, SONIC-16) is asserted by the AT9010 to the SONIC-16. During a register write cycle, this signal indicates a valid address on the bus. During a register read cycle, it indicates the SONIC-16 can begin sourcing data.

**SW-R** (AT9010, SONIC-16) is driven to the SONIC-16 to identify whether the current register access is a read or write cycle.

**RDY0** (SONIC-16, AT9010) is driven after the system has accessed the SONIC-16's registers and the SONIC-16 has completed the I/O cycle. The SONIC-16 may use this signal to insert wait states in the cycle.

The following signals are driven during AT memory cycles to the boot EPROM:

**EPROMRD** (AT9010, EPROM) is the chip enable for the boot EPROM.

**HAENB** (Comp, AT9010) is a signal which indicates when the BIOS EPROM is being accessed. It is the output of a comparator which uses the base address of the EPROM (in CR2, bits <5..2>) and LA<23..17> as inputs for an address decode.

**MEMR** (AT bus, AT9010) is input to the AT9010 to indicate a memory read cycle during EPROM memory access.

#### MASTER CYCLE SIGNALS

A description of master signals generated while requesting the bus during memory read and write cycles is given below. Again, the format followed is SIGNAL (ORIGIN, DESTINATION).

**HOLD** (SONIC-16, AT9010 and GAL) is the SONIC-16's DMA request signal that notifies the AT9010 that the SONIC-16 is requesting the bus.

**DRQx** (AT9010, AT bus) is the conversion of HOLD. It is driven by the AT9010 through the AT bus to the DMA controller.

**DACKx** (AT bus, AT9010) is the DMA acknowledgment signal which grants the DMA controller ownership of the AT bus.

**MASTER** (AT9010, AT bus) is asserted to the AT bus when DACKx is received. It disables the DMA buffers off the AT bus.

**HLDAAT/HLDA SONIC** (AT9010 to GAL to SONIC-16) is generated by the AT9010 when it has been granted ownership of the bus. Due to bugs in the original AT9010, HLDAAT is extended by the GAL up to one clock before being driven to the SONIC-16 as HLDASONIC. For the AT9010B, the problem is corrected and HLDA is driven directly to the SONIC-16.

**USER\_PIN1/PRE-EMPT** (AT9010, GAL) drives a 1 (or 0) to the GAL to enable (or disable) SONIC pre-emption. This is for the original AT9010 only.

The following signals are driven during the memory read or write cycles of master mode operation:

**BALE** (AT bus, AT9010) is high for the duration of master mode.

**ADS** (SONIC-16, AT9010) is an address strobe driven by the SONIC-16 which notifies the AT9010 of a valid address on the bus.

**HAOE** (AT9010, A-buffers) is an enable to the address buffers which gate the address from the adapter card to the AT bus.

**MW-R** (SONIC-16, AT9010) is input to the AT9010 by the SONIC-16 to indicate a read operation (signal low) or a write operation (signal high).

**MEMR** (AT9010, AT bus) is an AT9010 conversion of the signal MW-/R and indicates a SONIC-16 read cycle of system memory.

**MEMW** (AT9010, AT bus) is an AT9010 conversion of the signal MW-R and indicates a SONIC-16 write cycle to system memory.

**SBHE** (AT9010, AT bus) and (AT bus, AT 9010) denotes data on the most significant byte  $D < 15..8 >$  of the data bus. It notifies the system bus during a SONIC-16 memory write and notifies the AT9010 during a SONIC-16 memory read.

**HDDIR** (AT9010, D-Buffers) is an input to the data buffers which identifies the direction of a data transfer. It drives high for data transfers to the AT bus during a memory write cycle. It drives low for data transfers to the adapter card during a memory read cycle.

**HD0E1** (AT9010, D-buffer) enables the data buffer for the upper byte of data  $D < 15..8 >$ .

**HD0E0** (AT9010, D-buffer) enables the data buffer for the lower byte of data  $D < 7..0 >$ .

**IOCHRDBUS/IOCHRDIAT** (AT bus to GAL to AT9010) are input and output signals for both the AT bus and the AT9010. For the original AT9010, the GAL translates the AT bus's IOCHRDBUS into IOCHRDIAT and drives this signal to the AT9010 to insert wait states and complete a memory access. For the AT9010B, the AT bus drives IOCHRDI directly to the AT9010B.

**RDYI** (AT9010, SONIC-16) indicates to the SONIC-16 that a memory cycle has completed. The SONIC-16 will wait for this signal before re-asserting ADS to begin another cycle.

## HUB MANAGEMENT SIGNALS

The following signals are generated during use of the in the hub management interface. The signals are presented in the format of SIGNAL (ORIGIN, DESTINATION) or SIGNAL (ORIGIN to DESTINATION / ORIGIN to DESTINATION).

**MCRS/CRS** (IR/M bus to BTL to SONIC-16) is the management carrier sense which indicates data on the SONIC-16's receive lines.

**MRXD/RXD** (IR/M bus to BTL to SONIC-16) is the management receive data.

**MRXC/RXC** (IR/M bus to BTL to SONIC-16) is the management receive clock.

**PCOMP** (SONIC-16 to BTL to M bus) is the SONIC-16's packet compression output pin which causes the transmitting RIC to inhibit the MRXC clock upon mismatch of the packet's destination address with the SONIC-16's CAM when the SONIC-16 is in managed-hub mode.

**TXE/IRE** (SONIC-16 to BTL to IR bus) is the SONIC-16's transmit enable signal.

**TXD/IRD** (SONIC-16 to BTL to IR bus) is the SONIC-16's transmit data.

**TXC/IRD** (Flip-flop to BTL and SONIC-16 to IR bus) is a 10 MHz transmit clock signal.

**ACKI** (IR bus, PAL) passes permission (ACKI = 1) or denial (ACKI = 0) to the SONIC-16 from the RIC above it in the arbitration chain. (This is for transmission arbitration.)

**ACKO** (PAL, IR bus) passes permission (ACKO = 1) to transmit over the Inter-RIC bus to the RIC below the SONIC-16 if the SONIC-16 has permission to transmit and does not want to transmit. ACKO passes denial (ACKO = 0) if the SONIC-16 does not have permission (ACKI = 0) or the SONIC-16 wants to transmit (ACKI = 1).

**TX\_EN** (PAL, BTL) is the transmit drive enable of the BTL. It is asserted when the SONIC-16 transmits (TXE = 1), has permission to transmit (ACKI = 1) and is configured for an external ENDEC (EXT = 1).

**ACTNd** (PAL, BTL) notifies the RICs that the SONIC-16 wants to transmit. It is asserted when the SONIC-16 transmits (TXE = 1) and it has permission to transmit (ACKI = 1).

**ANYXNd** (PAL, BTL) is asserted when the SONIC-16 transmits (TXE = 1) and it does not have permission to transmit (ACKI = 0). ANYXNd indicates a collision on the Inter-RIC bus.

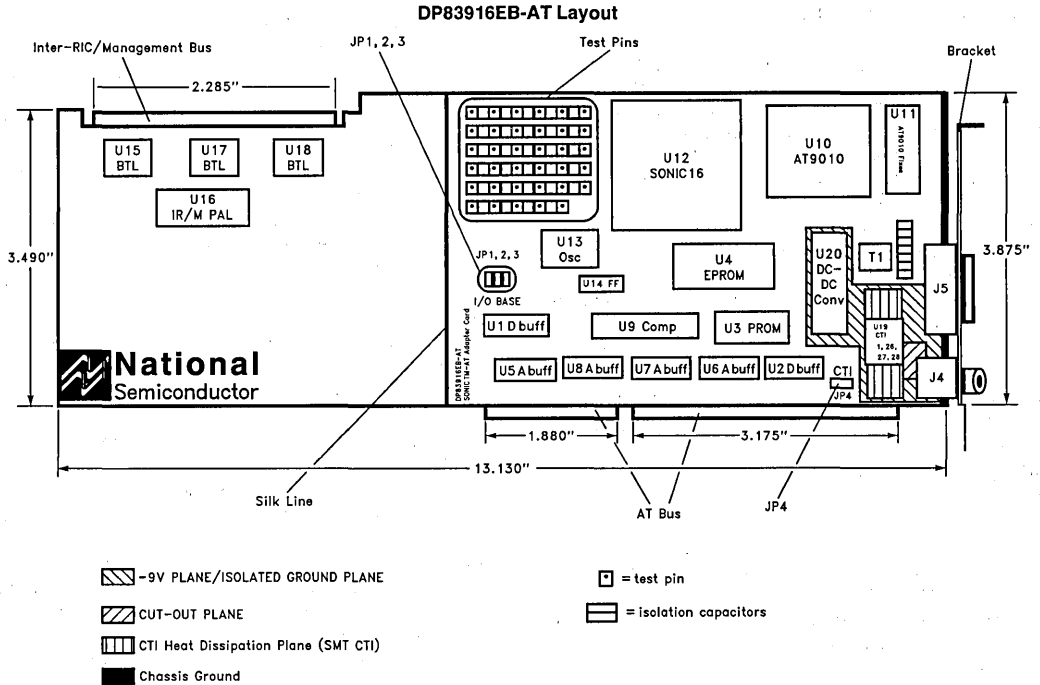
**ANYXNs** (BTL, PAL) senses transmit collisions on the Inter-RIC bus and the network.

**COL** (PAL, SONIC-16) is driven by the PAL when there is a transmit collision on the Inter-RIC bus (ANYXN = 1) or there is a receive collision on the network (COLN = 1) and during either event the SONIC-16 is transmitting (TXE = 1).

**COLN** (IR bus to BTL to PAL) indicates receive collisions on the network.

**APPENDIX V: DP83916EB-AT CARD AND COMPONENT PLACEMENT**

This appendix illustrates the placement of the DP83916EB-AT components. Special layout considerations for the DP8392 (Coaxial Transceiver Interface) are identified. (Details regarding these considerations can be found in the data sheet for the DP8392.) The silk line illustrates a place on the board in which power and ground planes and Inter-RIC/Management signals are non-overlapping; hence, the full card can be cut into a half card along this line. JP1-JP3 and JP4 are located on the adapter card as shown. A bed of test pins is provided on the card. The test pin signals are defined in Appendix VI.



TL/F/11707-14



**APPENDIX VI: TEST PIN LAYOUT**

The test pins and their associated signals are presented below. Most other signals can be probed off the bus via an AT extender card.

|                    |                           |                              |                             |                              |                     |
|--------------------|---------------------------|------------------------------|-----------------------------|------------------------------|---------------------|
| TP1<br>●<br>HOLD   | TP2<br>●<br>HLDASONIC     | TP3<br>●<br>MW/ $\bar{R}$    | TP4<br>●<br>IOCHRDYAT       | TP5<br>●<br>$\bar{CS}$       | TP6<br>●<br>SAS     |
| TP7<br>●<br>INT    | TP8<br>●<br>BSCK          | TP9<br>●<br>$\bar{RDY}_i$    | TP10<br>●<br>$\bar{RDY}_o$  | TP11<br>●<br>S2              | TP12<br>●<br>S1     |
| TP13<br>●<br>S0    | TP14<br>●<br>COL          | TP15<br>●<br>RXD             | TP16<br>●<br>CRS            | TP17<br>●<br>RXC             | TP18<br>●<br>TXE    |
| TP19<br>●<br>TXC   | TP20<br>●<br>TXD          | TP21<br>●<br>$\bar{PCOMP}$   | TP22<br>●<br>$\bar{ADS}$    | TP23<br>●<br>SW/ $\bar{R}$   | TP24<br>●<br>RA5    |
| TP25<br>●<br>RA4   | TP26<br>●<br>RA3          | TP27<br>●<br>V <sub>CC</sub> | TP28<br>●<br>$\bar{HDOE}_1$ | TP29<br>●<br>$\bar{HDOE}_0$  | TP30<br>●<br>GROUND |
| TP31<br>●<br>HDDIR | TP32<br>●<br>$\bar{HAOE}$ | TP33<br>●<br>$\bar{RESET}$   | TP34<br>●<br>$\bar{HAENB}$  | TP35<br>●<br>$\bar{POSCS}_3$ |                     |

TL/F/11707-15

## APPENDIX VII: DP83916EB-AT DESIGN CHANGE RECOMMENDATION

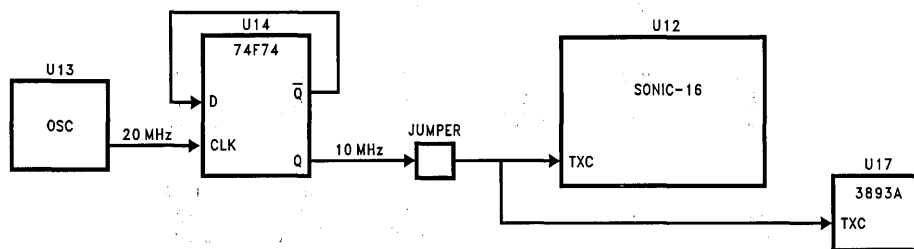
This appendix outlines a design change recommendation for future designs which implement both hub management and alternative media on the same card. There is a signal TXC which is driven from the flip-flop (U14) to the TXC pin (pin 12) of the SONIC-16 (U12) and the Inter-RIC/Management transmit BTL (U17). This signal is intended to provide a 10 MHz transmit clock to the SONIC-16's MAC when the SONIC is configured for hub management and hence, external ENDEC mode.

The current design of the DP83916EB-AT, however, provides TXC whether the SONIC-16 is in internal ENDEC mode (as a stand-alone node) or external ENDEC mode (as a hub manager). When the SONIC-16 is in internal ENDEC mode, it drives a 10 MHz signal (from the ENDEC) out of the chip. Because the flip-flop is also driving this node, a problem could arise if the clocks become out of phase.

Although no problems have arisen in lab testing, it is recommended that one of the following changes be made on future designs implementing the same functionality as the DP83916EB-AT. One of these changes will be implemented on the next version of this adapter card.

1. Place a jumper between the flip-flop output and the SONIC-16's TXC pin. Populate the jumper during hub management mode only.

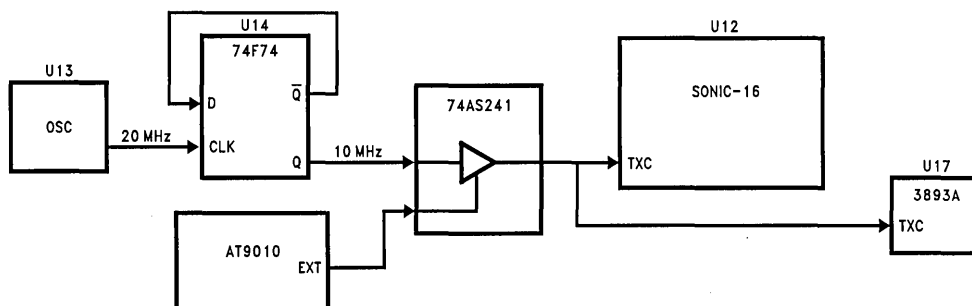
### Jumper Solution



TL/F/11707-16

2. Place a TRI-STATE® buffer (74AS241A) between the flip-flop and the SONIC-16's TXC pin. Use the USER\_PIN3 EXT from the AT9010 as the buffer enable. When EXT is 0 for internal ENDEC, the buffer will be disabled. When EXT is 1 for external ENDEC, the buffer will be enabled and the 10 MHz signal will drive into the SONIC-16's TXC pin and the transmit BTL.

### TRI-STATE Buffer Solution



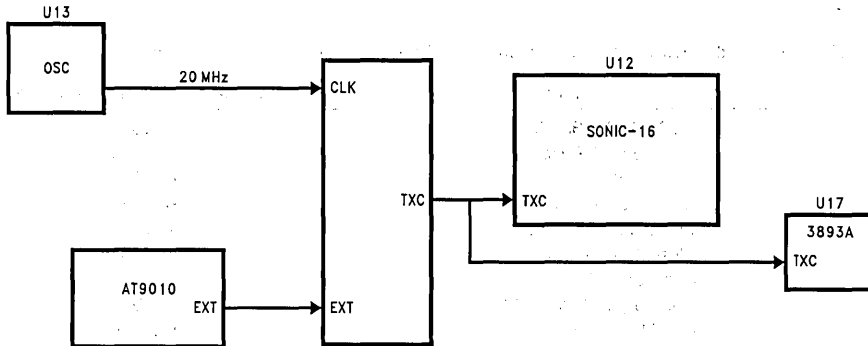
TL/F/11707-17

3. Delete the flip-flop (U14) and replace the PAL (U11) with a registered GAL (GAL16V8). Inout the 20 MHz clock into the GAL's clock pin (pin 1). Define TXC in the pin assignment as TXC (pin 15). Change the device declaration to "P16V8R". Include in the GAL equations the following two equations:  $TXC := !TXC$  and  $TXC.OE = EXT$ . The signal TXC will only drive out when EXT is 1 for external ENDEC mode. When EXT is 0, TXC will be TRI-STATE.

**Note:** The ":"=" defines that the equation for TXC is clocked on the rising edge of the 20 MHz signal. Since all equations in the GAL are asynchronous except for the TXC equation, there should only be a colon before the equal sign for the TXC equation.

This solution reduces the overall chip count by one and places all hub management signals in one IR/M GAL. However, because the 20 MHz and 10 MHz signals are traversing half the length of the card, radiation of noise is increased.

### Inter-RIC/Management GAL Solution



TL/F/11707-18

Additional Equations in GAL:

$TXC := !TXC$   
 $TXC.OE = EXT$

**APPENDIX VIII: COMPATIBILITY TESTING**

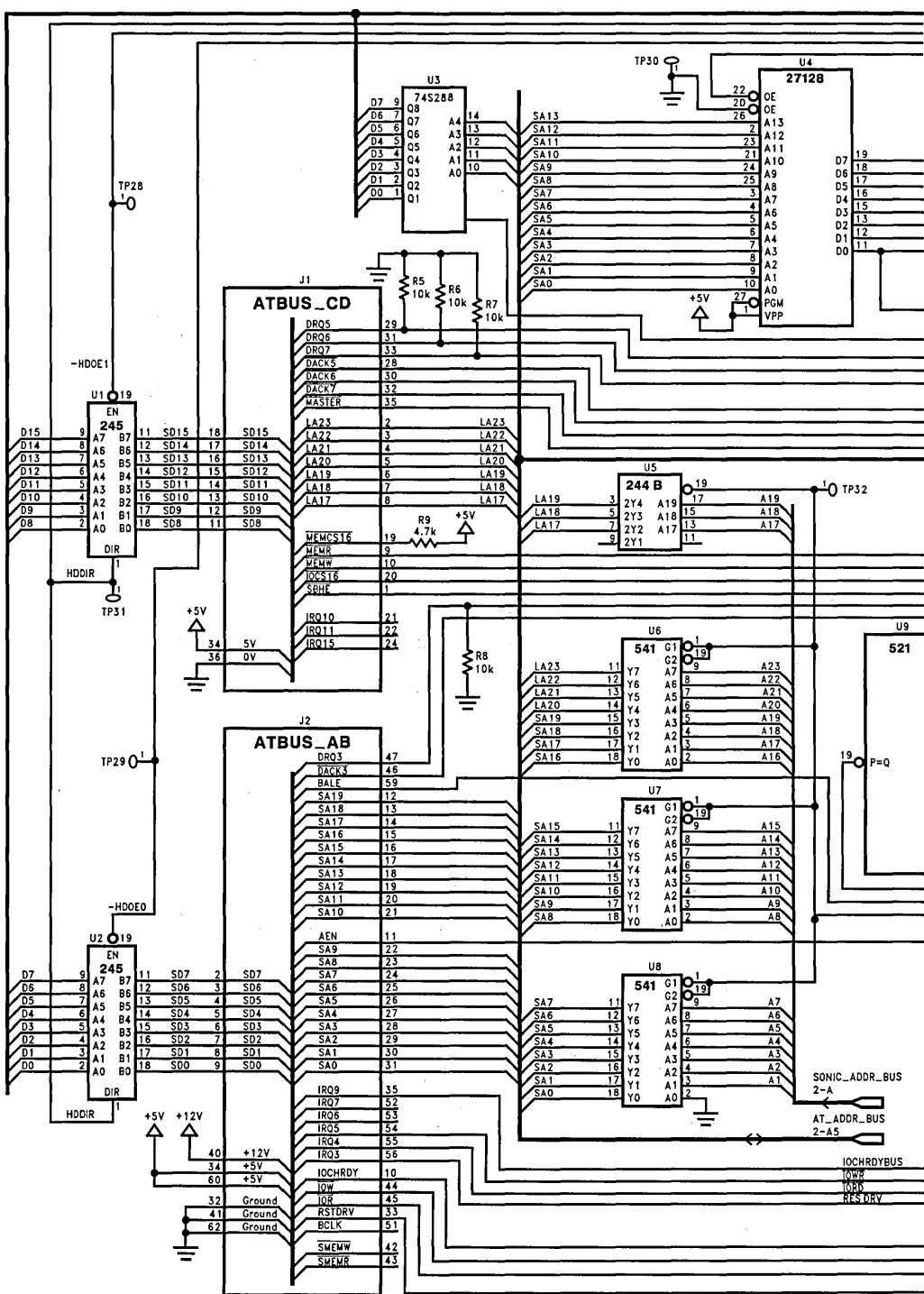
This appendix describes basic compatibility testing results for the DP83916EB-AT.

The DP83916EB-AT has been tested in various PC-AT/Compatible and EISA machines. The original AT9010 chip was used, SONIC-16 pre-emption was disabled and the master data transfer rate was 5 MB/s. The following basic tests were used:

1. Initialization and loopback
2. CAM load and Ethernet address PROM read
3. 16k EPROM enable/read
4. Simultaneous transmission/reception in two-node network
5. Continuous manual CTI loopback (set RCR to FE00h, set CTDA link field and CTDA register to current TDA address, set CR to 2h)

The DP83916EB-AT passed the basic tests in the following PC-AT/Compatible and EISA machines:

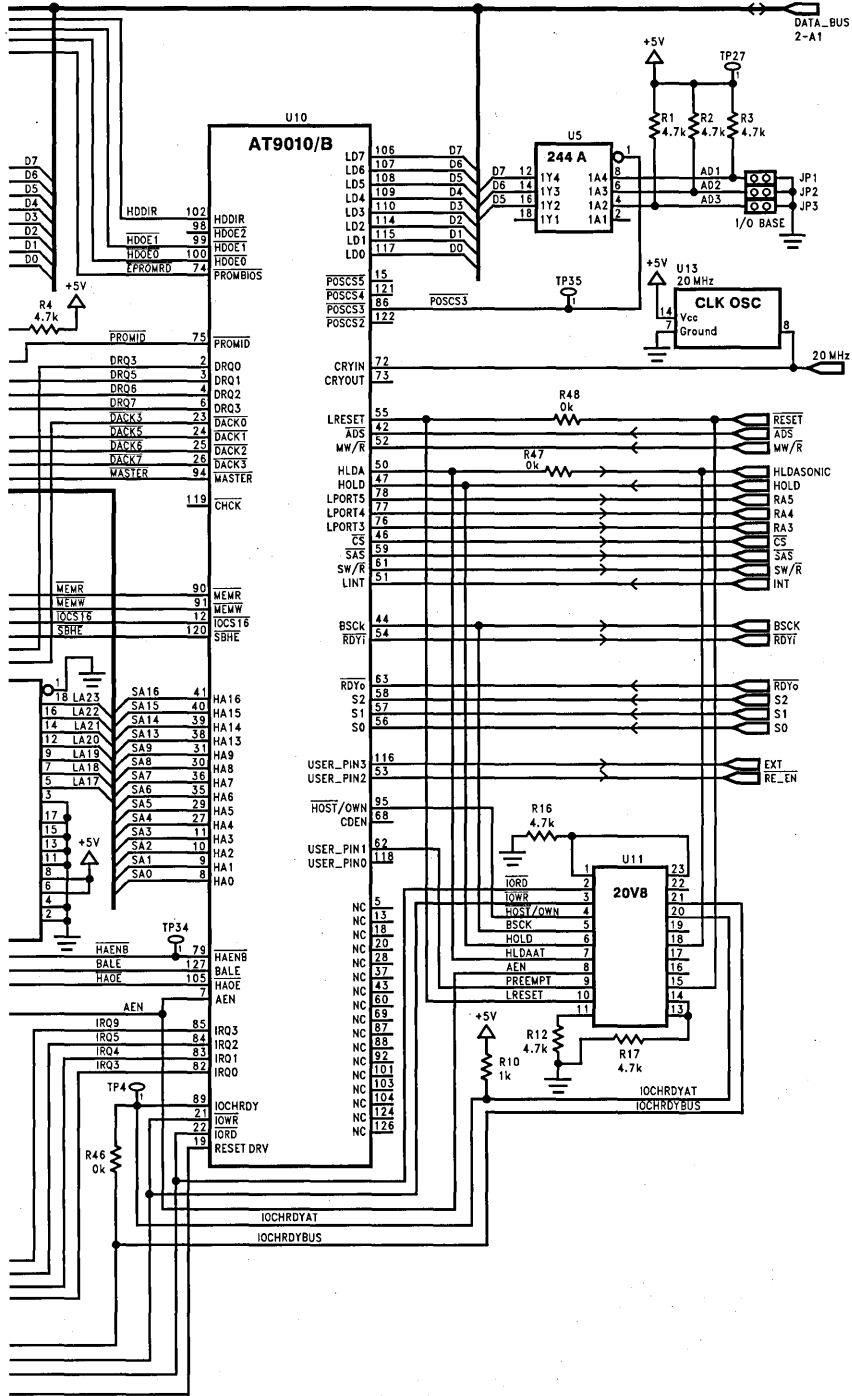
|                          | <b>Machine</b>      | <b>Fastest Master Transfer Speed</b> |
|--------------------------|---------------------|--------------------------------------|
| AT:                      | ALR 386DX/33 MHz    | 8 MB/s                               |
|                          | AST 386/33 MHz      | 8 MB/s                               |
|                          | Clone 386           | 8 MB/s                               |
|                          | Clone 386SX/16 MHz  | 6.7 MB/s                             |
|                          | Clone 386/25 MHz    | 10 MB/s                              |
|                          | Clone 386/33 MHz    | 10 MB/s                              |
|                          | Compaq 286          | 8 MB/s                               |
|                          | Compaq 486DX/50 MHz | 6.7 MB/s                             |
|                          | Dell 386/25 MHz     | 6.7 MB/s                             |
|                          | Dell 486SX/20 MHz   | 6.7 MB/s                             |
|                          | Dell 486DX/50 MHz   | 8 MB/s                               |
|                          | Everex 386SX/16 MHz | 10 MB/s                              |
|                          | Everex 386/33 MHz   | 10 MB/s                              |
|                          | Zeos 486/33M Hz     | 10 MB/s                              |
|                          | EISA:               | ALR EISA 386DX/33 MHz                |
| AST EISA 486/33 MHz      |                     | 8 MB/s                               |
| Compaq EISA 386/33 MHz   |                     | 6.7 MB/s                             |
| Compaq EISA 486SX/25 MHz |                     | 6.7 MB/s                             |
| Compaq EISA 486DX/50 MHz |                     | 10 MB/s                              |
| Dell EISA 486/25 MHz     |                     | 8 MB/s                               |
| Dell EISA 486DX/33 MHz   |                     | 8 MB/s                               |
| HP EISA 486/33 MHz       |                     | 8 MB/s                               |
| NEC EISA 386/33 MHz      |                     | 8 MB/s                               |



AT Bus Interface

TL/F/11707-19

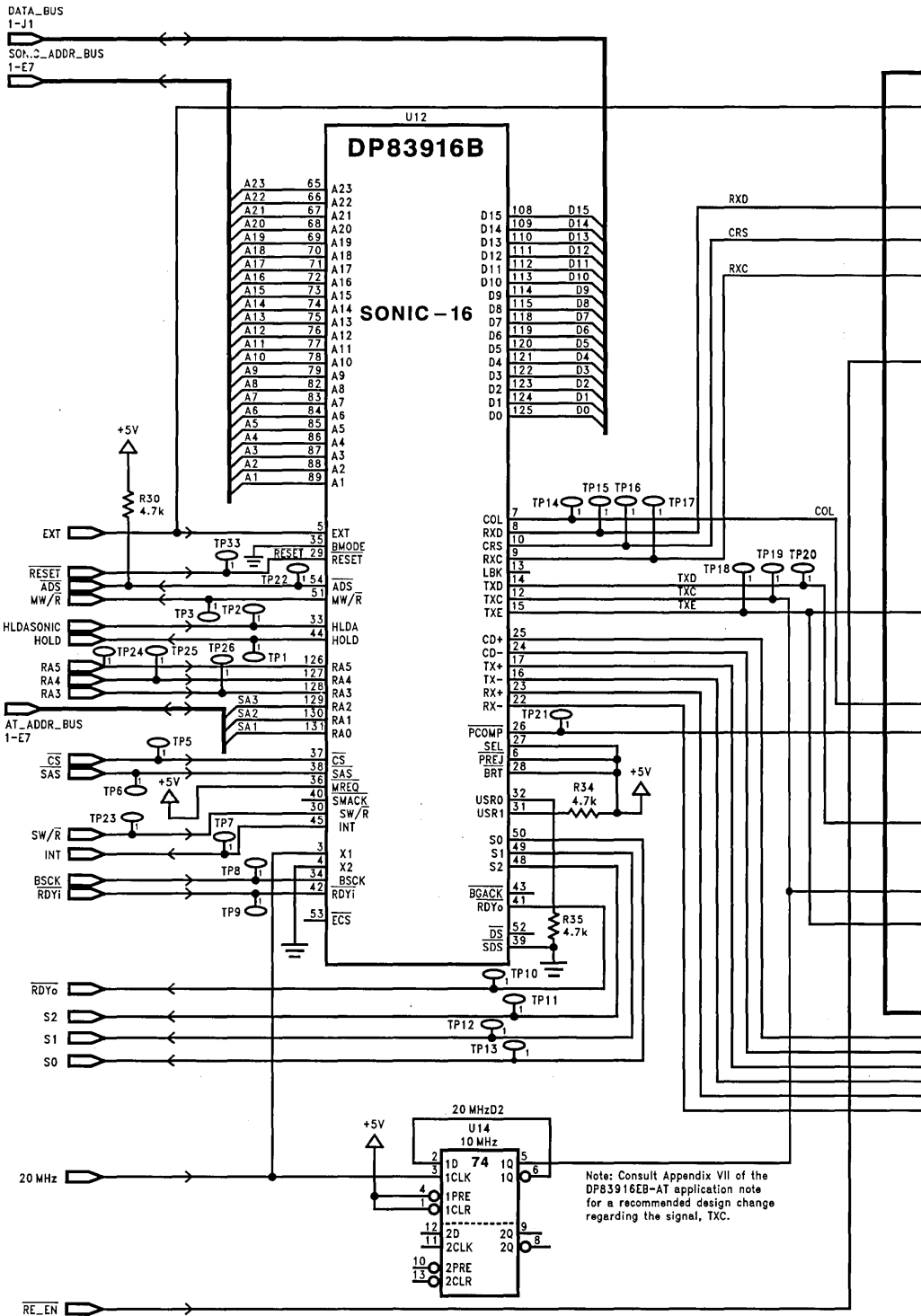




Note: For original AT9010, install U11 and leave R46, R47 and R48 open. For AT9010B install R46, R47 and R48 and leave U11 open.

TL/F/11707-20

AT Bus Interface (Continued)

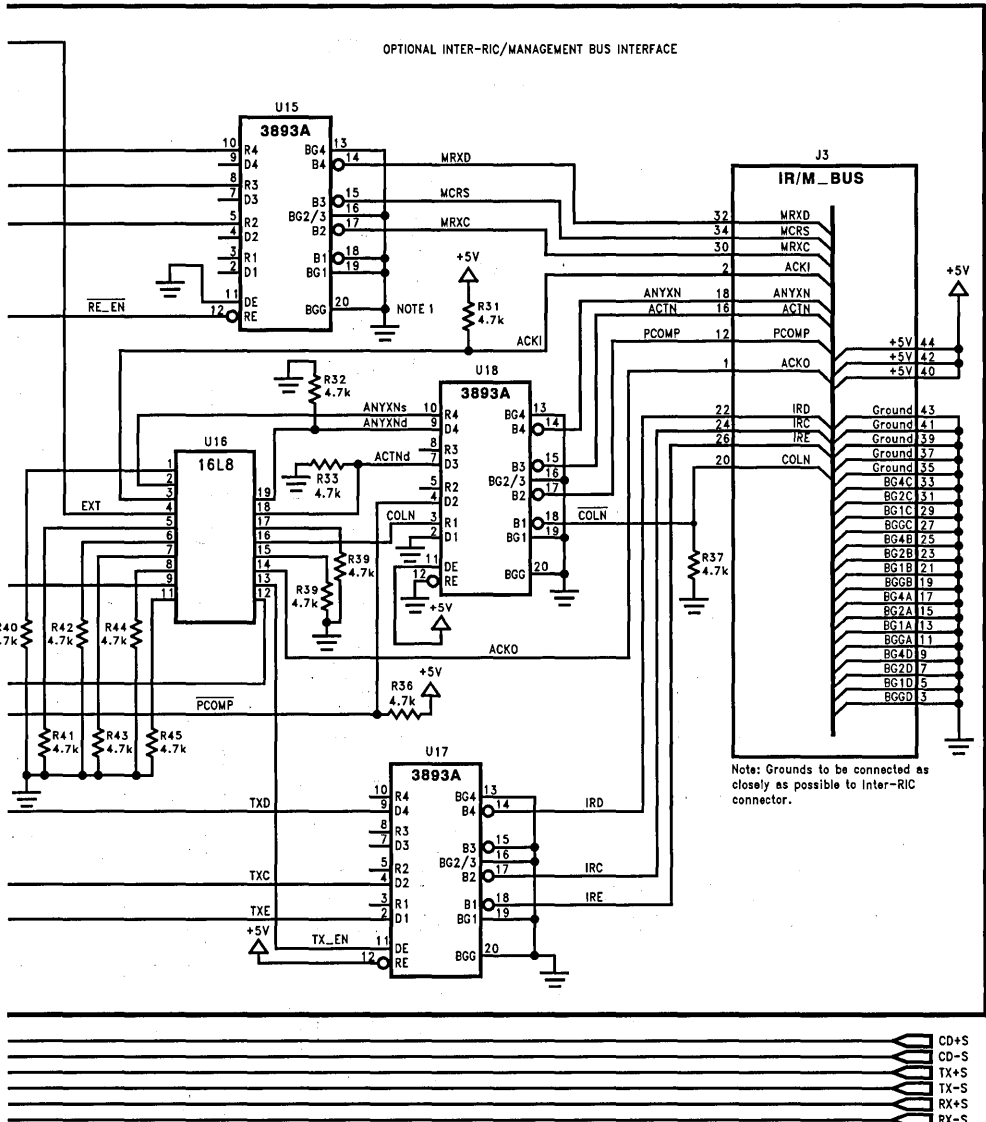


Note: Consult Appendix VII of the DP83916EB-AT application note for a recommended design change regarding the signal, TXC.

**SONIC and Inter-RIC/Management Interface**

TL/F/11707-21

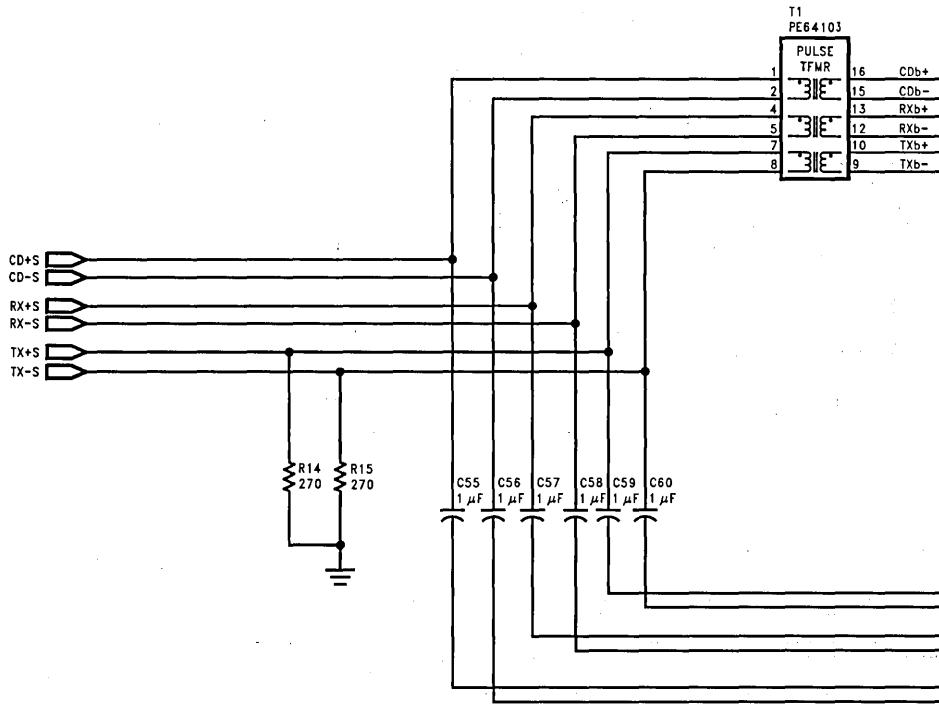




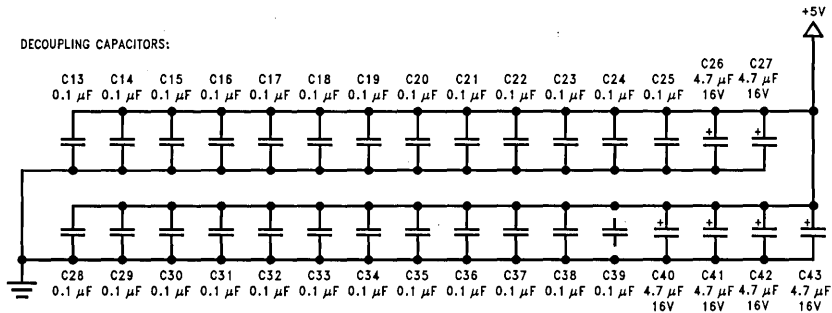
SONIC and Inter-RIC/Management Interface (Continued)

TL/F/11707-22

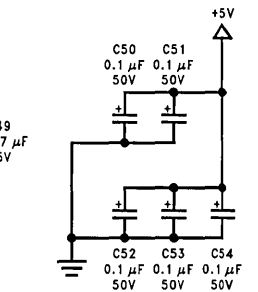
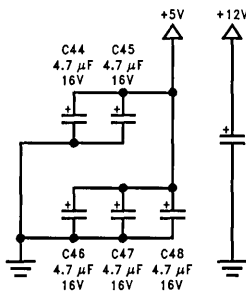
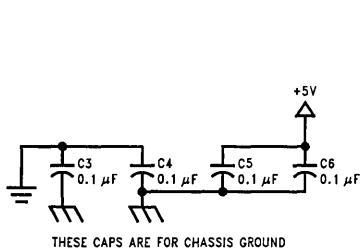




DECOUPLING CAPACITORS:

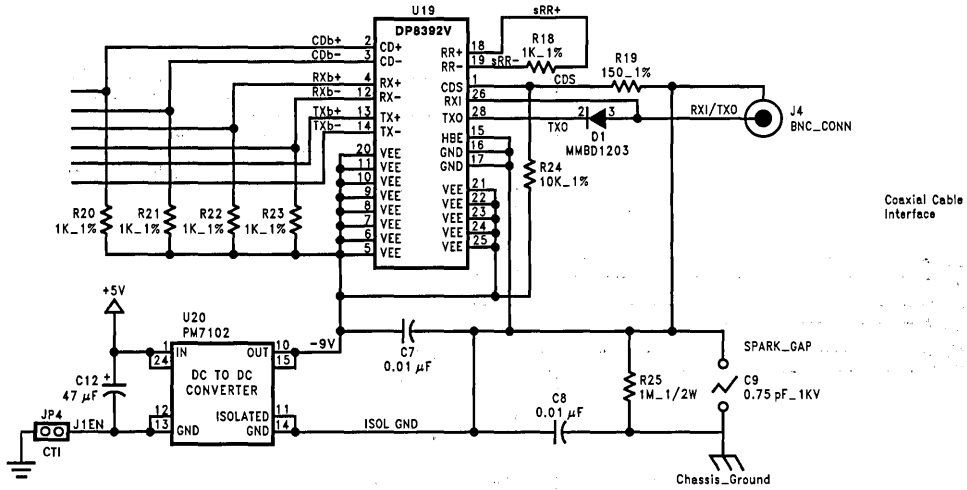


ONE 0.1 μF FOR EACH IC, THREE 0.1 μF FOR SONIC-16 AND AT9010, AND 4.7 μF FOR EVERY FOUR IC'S

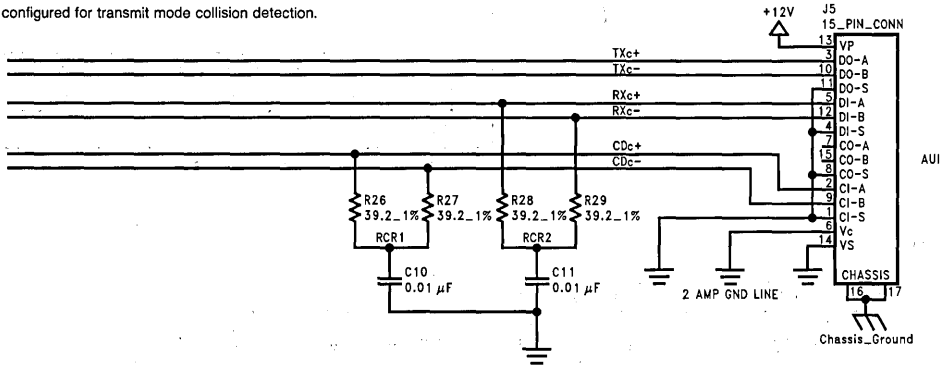


Media Interface

TL/F/11707-23



Note: CTI configured for transmit mode collision detection.



Note: Unless otherwise specified, resistors are 5%, 1/4W.

Media Interface (Continued)

TJ/F/11707-24

# DP83932EB-EISA SONIC/ EISA Packet Driver for PC/TCP

National Semiconductor  
Application Note 859



AN-859

## INTRODUCTION

This is a complete program listing for a network device driver for the DP83932EB-EISA SONIC EISA Demonstration and Evaluation Board. This driver enables the DP83932-EISA to operate with the Personal Computer-based TCP/IP software package, distributed by FTP Software Inc., called PC/TCP. Contact FTP Software Inc. at (617) 246-0900 for more information about the PC/TCP product offerings.

This driver conforms to version 1.9 of FTP Software's Packet Driver Specification, and works with version 2.x of the PC/TCP product (and products that have adopted the Packet Driver Specification).

This program listing is provided as an example of drive software for the DP83932 Systems Oriented Network Interface Controller (SONIC). The driver is written in Microsoft C (5.1 or greater) and Microsoft Assembler (5.1 or greater). Since the majority of the software is written in C, the concepts provided are easily portable to other environments.

This example software is provided as an example, and is not necessarily the most optimal implementation. The code has been thoroughly tested with PC/TCP.

The driver is listed by modules in the following order:

1. pktdrv.c
2. far.c
3. isr.c
4. sonic.c
5. pktdrv.h
6. sonic.h
7. isrlib.asm
8. pktint.asm
9. makefile

## PKTDRV.C

```

static char pktdrv_rcsid[]="@(#) $ID:$";
/*
*****
*           Copyright (c) 1992 by National Semiconductor Corporation           *
*           All Rights Reserved   *
*****
=====
FILE:      pktdrv.c
NOTES:    This program is a packet driver that provides a common interface
          between PC/TCP's kernel and NSC's SONIC hardware. This program
          was based on a set of drivers provided by Clarkson from FTP.
=====
UPDATE LOG:
When/Who           Why/What/Where
-----
11/30/90 Mike Lui           Convert to work for SONIC 32 bit
04/10/92 Michael Zhang     Added read_config();
                              Added 'transmitactive=1' in send_packet();
=====
*/

#include <stdio.h>
#include <dos.h>
#include <memory.h>
#include <string.h>
#include "pktdrv.h"
#include "sonic.h"

/* externals */
extern void (interrupt far drv_isr)(); /* the interrupt we use */
extern unsigned _psp; /* segment address of PSP */

/* Driver information */
static unsigned int   drv_version = 1; /* driver version */
static unsigned char  drv_class = 1; /* driver class */
static unsigned int   drv_type = 14; /* driver type */
static unsigned char  drv_number = 0; /* driver number */
static unsigned int   drv_funct = 5; /* basic and high-
                                     performance driver function */

static char drv_name[] = /* driver name */
    "National Semiconductor SONIC/TCP 32-bit Packet Driver";
static char cpy_msg[] =
    "Copyright (c) 1992 by National Semiconductor Corporation";
static char drv_rev[] = "1.2"; /* current driver rev */
static unsigned char BOARD_ID[]={ 0x41,0x98,0x10,0x01 }; /* PLX1001 */
static HANDLE handle_tbl[MAX_HANDLES]; /* create handle structs */
static void read_config(); /* read board config info */
void (interrupt far *sys_isr)(); /* remember system isr */
char far *pkt_signature = "PKT DRV";
unsigned int packet_int_no = 0x60; /* interrupt for communications */
static unsigned far *psp_ptr; /* pointer to PSP */
unsigned mem_sz; /* program memory size in paragraphs */
static void usage();

union REGS r_regs;
struct SREGS s_regs;
int send_pending; /* required for Synernetics */

```

TL/F/11720-1

```

static int syn_installed;          /* required for Synernetics */

extern int opterr;
extern int optind;
extern char *optarg;

/*
 * main()
 *
 * Main procedure.
 * Once initialization is complete terminate and stay resident.
 */
main(argc, argv)
int argc;
char *argv[];
{
    psp_ptr = (unsigned far *)((unsigned long)_psp << 16);
    mem_sz = (psp_ptr[1] - _psp);

    read_config();                /* read expansion board config*/

    init_drv(argc, argv);         /* initialize driver and hardware */

    outpw(regbase+cr, 8);         /* enable receiver */

    /* terminate and stay resident */
    _dos_keep(0, mem_sz);
}

/*
 * int_handler()
 *
 * This routine is called from an assembly isr routine "drv_isr"
 * to handle the application interrupt. The isr routine passes a
 * set of pointers of the registers to this routine. Register AH
 * contains which function is to be performed. These registers will
 * be restored in "drv_isr" before returning from the interrupt.
 *
 * Return values: If an error occurred the value will be in
 * the DH register and the carry bit of cflag
 * will be set.
 */
int_handler(regs, sregs)
union REGS far *regs;
struct SREGS far *sregs;
{
    int ret_val;

    switch(regs->h.ah) {
    case 1:
        ret_val = driver_info(regs, sregs);
        break;
    case 2:
        ret_val = access_type(regs, sregs);
        break;
    case 3:
        ret_val = release_type(regs, sregs);
        break;
    }
}

```

TL/F/11720-2

```

case 4:
    ret_val = send_packet(regs, sregs);
    break;
case 5:
    ret_val = terminate(regs, sregs);
    break;
case 6:
    ret_val = get_address(regs, sregs);
    break;
case 7:
    ret_val = reset_interface(regs, sregs);
    break;
case 10:
    ret_val = get_param(regs, sregs);      /* high-performance function */
    break;
case 11:
    ret_val = as_send_pkt(regs, sregs);    /* high-performance function */
    break;
case 24:
    ret_val = get_stats(regs, sregs);
    break;
default:
    ret_val = BAD_COMMAND;
}

if(ret_val) {
    regs->h.dh = ret_val;                  /* put error code into dh */
    regs->x.cflag |= 0x1;                  /* and set carry bit */
}

/*
 * driver_info()
 *
 * Return information on the driver interface. Handle is optional
 * and is not used in new driver??
 *
 * Return values: 0 - Success
 */
driver_info(regs, sregs)
union REGS far *regs;
struct SREGS far *sregs;
{
    regs->x.bx = drv_version;              /* driver version */
    regs->h.ch = drv_class;                /* driver class */
    regs->x.dx = drv_type;                 /* driver type */
    regs->h.cl = drv_number;               /* driver number */
    regs->x.si = (unsigned)drv_name;       /* driver name */
    sregs->ds = (unsigned long)((char far *)drv_name) >> 16;
    regs->h.al = drv_funct;                /* driver function */
    return 0;
}

/*
 * access_type()
 *
 * Initiate access to packets for the specific type. Since the packet
 * type field needs to have the bytes of 16 bit values swapped, the
 * handle will store the type field byte swapped.

```

TL/F/11720-3

```

*
*   Return values:  0 - Success
*                   >0 - Failure
*/
access type(regs, sregs)
union REGS far *regs;
struct SREGS far *sregs;
{
    int i, n,
        open_handle = OPEN,           /* available handle */
        type_cnt;

    /* first check a few things to make sure packet access is ok */

    /* check class */
    if(regs->h.al != drv_class) {
        return NO_CLAS;
    }

    /* check type (ours or generic) */
    if(!((regs->x.bx == drv_type) || (regs->x.bx == -1))) {
        return NO_TYPE;
    }

    /* check number (ours or generic) */
    if(!((regs->h.dl == 0) || (regs->h.dl == 1))) {
        return NO_NUMBER;
    }

    /* check packet type length, if too long its not ours */
    if(regs->x.cx > MAX_TYPE_LEN) {
        return TYPE_INUSE;
    }

    /*
     * now check for an available handle and if the handle already
     * exists with same packet type.
     */
    type_ptr = (char far *)(((unsigned long)sregs->ds << 16) | regs->x.si);

    for(i = 0; i < regs->x.cx; i++)
        type_buf[i] = type_ptr[i];

    for(n = 0; n < MAX_HANDLES; n++) {
        if(handle_tbl[n].in_use) {
            /* check packet type */
            type_cnt = MIN(regs->x.cx, handle_tbl[n].len);
            if(!far_memcmp((char far *)type_buf,
                (char far *)handle_tbl[n].type, type_cnt))
                return TYPE_INUSE; /* duplicate types */
        }
        else if(open_handle == OPEN)
            open_handle = n; /* grab first open handle */
    }

    if(open_handle == OPEN)
        return BAD_HANDLE; /* no available handles */

    /* copy the handle */
    handle_tbl[open_handle].in_use++;
}

```

TL/F/11720-4

1

```

for(i = 0; i < regs->x.cx; i++) {
    handle_tbl[open_handle].type[i] = type_buf[i];
}
handle_tbl[open_handle].len    = regs->x.cx;
handle_tbl[open_handle].rec_es = sregs->es;
handle_tbl[open_handle].rec_di = regs->x.di;

regs->x.ax = open_handle;          /* return handle */
return 0;                          /* return success */
}
/*
 * release_type()
 *
 * Release access to packets with a particular handle.
 *
 * Return values:  0 - Success
 *                >0 - Failure
 */
release_type(regs, sregs)
union REGS far *regs;
struct SREGS far *sregs;
{
    if(chk_handle(regs->x.bx))
        return BAD_HANDLE;

    /* release handle */
    handle_tbl[regs->x.bx].in_use = 0;
    return 0;
}

/*
 * send_packet()
 *
 * Send packet buffer.
 *
 * Return values:  0 - Success
 *                >0 - Failure
 */
send_packet(regs, sregs)
union REGS far *regs;
struct SREGS far *sregs;
{
    char far *frame_ptr;          /* pointer to frame */
    unsigned long pkt_addr;      /* physical address of packet */
    unsigned int buf_len;       /* frame length */
    int i;
    tda_struct *tmp_tda;
    short previous_tda;
    unsigned short addr;

    /* check if frame is too big */
    if(regs->x.cx > BUF_SZ) {
        return NO_SPACE;
    }

    /* update driver stats */
    drv_stats.packets_out++;
    drv_stats.bytes_out += regs->x.cx;
}

```



```

/* point to the app's send frame */
frame_ptr = (char far *)(((unsigned long)sregs->ds << 16) |
                          regs->x.si);
pkt_addr = (unsigned long) sregs->ds * 16 + regs->x.si;

buf_len = regs->x.cx;                               /* frame+FC+SNAP length */

/* save current tda */
previous_tda=curtda;

if (transmitactive) {
    /* network is currently busy transmitting, just queue up the tda */
    if (curtda==TDANUM-1)
        return CANT_SEND;
    else {
        /* copy data area from the frame */
        far_memcpy((char far *)&tba[curtda+1], &frame_ptr[0], regs->x.cx);
        addr=tda_addr+(curtda+1)*sizeof(tda_struct);
        tmp_tda=(tda_struct*)addr;
        tmp_tda->pkt_size=buf_len;
        tmp_tda->frag_count=1;
        tmp_tda->frag_size=buf_len;
        tmp_tda->link |= 1;
        tmp_tda->type = BASIC;
        addr-=sizeof(tda_struct);
        tmp_tda=(tda_struct*)addr;
        tmp_tda->link &= 0x0fffe;
        curtda++;
    }
}
else {
    /* network is free */
    retry=0;
    /* copy data area from the frame */
    far_memcpy((char far *)&tba[0], &frame_ptr[0], regs->x.cx);
    tmp_tda=(tda_struct*) tda_addr;
    tmp_tda->pkt_size=buf_len;
    tmp_tda->frag_count=1;
    tmp_tda->frag_size=buf_len;
    tmp_tda->link |= 1;
    tmp_tda->type = BASIC;
    tda_head=0;
    tda_tail=1;
    curtda=0;
    outpw(regbase+ctda, tda_start_addr); /* load ctda */
    transmitactive=1;
}

outpw(regbase+cr, 2);                               /* issue the transmit command */

return 0;
}

/*
 * terminate()
 *
 * Terminate the driver.
 *
 * Return values:  0 - Success
 *                >0 - Failure
 */

```

TL/F/11720-6

```

*/
terminate(regs, sregs)
union REGS far *regs;
struct SREGS far *sregs;
{
    int sonic_irq;

    sonic_irq=3;

    _dos_setvect(packet_int_no, sys_isr);          /* put back system isr */

    sonic_isr_disable(sonic_irq);                 /* remove sonic interrupt */
    /* free environment memory */
    _dos_freemem(psp_ptr[0x16]);

    /* free memory and return to app */
    if(_dos_freemem( psp))
        return CANT_TERMINATE;

    return 0;
}

/*
* get_address()
*
* Get the local net address.
*
* Return values:  0 - Success
*                >0 - Failure
*/
get_address(regs, sregs)
union REGS far *regs;
struct SREGS far *sregs;
{
    int i, old_mode;
    char far *addr_ptr;                          /* pointer to address */

    if(chk_handle(regs->x.bx))
        return BAD_HANDLE;

    /* get buffer */
    addr_ptr = (char far *)(((unsigned long)sregs->es << 16) | regs->x.di);

    /*
    * copy ethernet address from hardware.
    * regs->x.cx is the length of buffer, fail if address
    * is too big to fit in buffer - NO_SPACE
    */
    if(regs->x.cx < 6)
        return NO_SPACE;

    regs->x.cx = 6;

    for(i = 0; i < regs->x.cx; i++) {
        addr_ptr[i] = inp(regbase+0xc90+i);
    }

    return 0;
}

```

TL/F/11720-7

```

/*
 * reset_interface()
 *
 * Reset the interface for the particular handle. If more than one
 * handle is open return CANT_RESET so other applications (handles)
 * will not get confused.
 *
 * Return values:  0 - Success
 *                >0 - Failure
 */
reset_interface(regs, sregs)
union REGS far *regs;
struct SREGS far *sregs;
{
    char far *addr_ptr;                /* pointer to address */
    int      i, handle_cnt = 0;

    if(chk_handle(regs->x.bx))
        return BAD_HANDLE;

    /* check if there is more than one handle is open */
    for(i = MIN_HANDLE; i < MAX_HANDLES; i++)
        if(handle_tbl[i].in_use != 0) handle_cnt++;
    if(handle_cnt > 1)
        return CANT_RESET;

    /* Reset the hardware to a known state */
    /* Will need something maybe ??? */

    return 0;
}

/*
 * get_param()
 *
 * Return driver parameters
 *
 * Return values:  0 - Success
 *                >0 - Failure
 */
get_param(regs, sregs)
union REGS far *regs;
struct SREGS far *sregs;
{
    if(drv_funct !=5 && drv_funct != 6)
        return BAD_COMMAND;

    drv_param.major_rev=1;
    drv_param.minor_rev=9;
    drv_param.length=14;
    drv_param.addr_len=6;
    drv_param.mtu=1512;
    drv_param.multicast_aval=90;
    drv_param.rcv_bufs=3;
    drv_param.xmt_bufs=3;
    drv_param.int_num=0;
}

```

TL/F/11720-8

```

regs->x.di = (unsigned)&drv_param;          /* driver stats */
sregs->es = (unsigned long)((char far *)&drv_param) >> 16;

return 0;
}

/*
 * as_send_pkt()
 *
 * HIgh performance send packet.
 *
 * Return values:  0 - Success
 *                >0 - Failure
 */
as_send_pkt(regs, sregs)
union REGS far *regs;
struct SREGS far *sregs;
{
    char far *frame_ptr;                /* pointer to frame */
    unsigned long pkt_addr;            /* physical address of packet */
    unsigned int buf_len;              /* frame length */
    int i;
    tda_struct *tmp_tda;
    short previous_tda;
    unsigned short addr;

    /* check if frame is too big */
    if(regs->x.cx > BUF_SZ) {
        return NO_SPACE;
    }

    /* update driver stats */
    drv_stats.packets_out++;
    drv_stats.bytes_out += regs->x.cx;

    /* point to the app's send frame */
    frame_ptr = (char far *)(((unsigned long)sregs->ds << 16) |
                             regs->x.si);
    pkt_addr = (unsigned long) sregs->ds * 16 + regs->x.si;

    buf_len = regs->x.cx;                /* frame+FC+SNAP length */

    /* save current tda */
    previous_tda=curtda;

    if (transmitactive) {
        /* network is currently busy transmitting, just queue up the tda */
        if (curtda==TDANUM-1) {
            xmt_upcall(CANT_SEND, (char far *) &frame_ptr,regs->x.di,sregs->es);
            return CANT_SEND;
        }
        else {
            /* copy data area from the frame */
            far_memcpy((char far *)&tba[curtda+1], &frame_ptr[0], regs->x.cx);
            addr=tda_addr+(curtda+1)*sizeof(tda_struct);
            tmp_tda=(tda_struct*)addr;
            tmp_tda->pkt_size=buf_len;
            tmp_tda->frag_count=1;
        }
    }
}

```

```

        tmp_tda->frag_size=buf_len;
        tmp_tda->link |= 1;
        tmp_tda->type = HIGH_PERFORMANCE;
        tmp_tda->buffer=frame_ptr;
        tmp_tda->xmt_es=sregs->es;
        tmp_tda->xmt_di=regs->x.di;
        addr=sizeof(tda_struct);
        tmp_tda=(tda_struct*)addr;
        tmp_tda->link &= 0x0fffe;
        curtda++;
        tda_tail=curtda+1;
    }
}
else {
    /* network is free */
    retry=0;
    /* copy data area from the frame */
    far_memcpy((char far *)&tba[0], &frame_ptr[0], regs->x.cx);
    tmp_tda=(tda_struct*) tda_addr;
    tmp_tda->pkt_size=buf_len;
    tmp_tda->frag_count=1;
    tmp_tda->frag_size=buf_len;
    tmp_tda->link |= 1;
    tmp_tda->type = HIGH_PERFORMANCE;
    tmp_tda->buffer=frame_ptr;
    tmp_tda->xmt_es=sregs->es;
    tmp_tda->xmt_di=regs->x.di;
    curtda=0;
    tda_head=0;
    tda_tail=1;
    outpw(regbase+ctda, tda_start_addr); /* load ctda */
}

outpw(regbase+cr, 2); /* issue the transmit command */

return 0;
}

/*
 * get_stats()
 *
 * Return driver statistics.
 *
 * Return values:  0 - Success
 *                >0 - Failure
 */
get_stats(regs, sregs)
union REGS far *regs;
struct SREGS far *sregs;
{
    if(chk_handle(regs->x.bx))
        return BAD_HANDLE;

    regs->x.si = (unsigned)&drv_stats; /* driver stats */
    sregs->ds = (unsigned long)((char far *)&drv_stats) >> 16;

    return 0;
}

```

TL/F/11720-10

```

/*
 * drv_rcvr()
 *
 * Receiver procedure. Once a frame is recieved, we need to make two upcall
 * with the receiving routine provided by the application. The first
 * call (AX == 0) is to request a buffer to copy the frame to. The second
 * call (AX == 1) indicates that the frame has been copied.
 *
 * Return values:  0 - Success
 *                >0 - Failure
 */
/* void far drv_rcvr() */
drv_rcvr()
{
    int i;
    int handle_found = OPEN;          /* set if valid frame recieved */
    char far *cp_ptr;
    unsigned short addr;
    unsigned char far *frame;

    /* get the frame */
    while ((unsigned short)cur_rda->status != 0) {
        frame=(unsigned char far *) (((unsigned long) cur_rda->pkt_ptr1 << 28) |
                                     (unsigned short) cur_rda->pkt_ptr0);
        /* validate the received frame */
        for(i = MIN_HANDLE; i < MAX_HANDLES; i++) {
            if((handle_tbl[i].in_use == 0) ||
               (((unsigned long)handle_tbl[i].rec_es << 16) |
                handle_tbl[i].rec_di) == 0))
                continue;          /* go to next handle */
            if(!far_memcmp((char far *)handle_tbl[i].type,
                           &frame[ETYPE_OFS], handle_tbl[i].len)) {
                handle_found = i;
                break;
            }
        }
        if(handle_found == OPEN) {
            drv_stats.packets_dropped++;
            free_rda();
            continue;
        }
        if ((unsigned short) cur_rda->status & 0x0c) {
            drv_stats.packets_dropped++;
            free_rda();
            continue;
        }
    }

    /* update driver stats */
    drv_stats.packets_in++;
    drv_stats.bytes_in += (unsigned short) cur_rda->byte_count;

    /* first upcall, tell them frame size */
    app_rcv(0,handle_found, MAX((unsigned short) cur_rda->byte_count-4,64),
            (char far *)&cp_ptr, handle_tbl[handle_found].rec_di,
            handle_tbl[handle_found].rec_es);

    /* check if copy is permitted */

```

TL/F/11720-11

```

    if(cp_ptr == NULL) {
        drv_stats.packets_dropped++;
        free_rda();
        continue;
    }

    /* copy the frame */
    far_memcpy(&cp_ptr[0], &frame[0], (unsigned short)cur_rda->byte_count-4)

    /* second upcall, tell them frame has been copied */
    app_rcv(1, handle_found, (unsigned short) cur_rda->byte_count-4,
            (char far *)&cp_ptr,
            handle_tbl[handle_found].rec di,
            handle_tbl[handle_found].rec_es);

    /* free rda */
    free_rda();
}
return 0;
}

/*
 * free_rda()
 *
 * This routine is to free up the currently examined rda for later use
 *
 */

free_rda()
{
    static int first;
    unsigned short tmp_value;
    unsigned short addr;
    rda_struct * p_rda;

    /* check fifo overrun */
    if (inpw(regbase+isr) & ISR_RFO)
        outpw(regbase+isr, ISR_RFO);

    /* reinitialize the rda */
    cur_rda->status=0;
    cur_rda->byte_count=0;
    cur_rda->pkt_ptr0=0;
    cur_rda->pkt_ptr1=0;
    cur_rda->in_use=0x0ffff;
    cur_rda->pkt_link |= 1;

    /* link the previous rda to the current rda */
    if (currda==0) {
        addr=rda_start_addr+(RDANUM-1)*sizeof(rda_struct);
        p_rda=(rda_struct*) addr;
        p_rda->pkt_link&=0x0fffe;
    }
    else {
        addr=c_rda-sizeof(rda_struct);
        p_rda=(rda_struct*) addr;
        p_rda->pkt_link&=0x0fffe;
    }
}

```

TL/F/11720-12

```

    }

    /* get the first buffer number */
    if (!first) {
        previous_seqno=(unsigned short)cur_rda->seq_no >> 8;
        first=1;
    }

    /* check whether rba can be reused */
    if ((unsigned short)cur_rda->seq_no >> 8 != previous_seqno) {
        previous_seqno=(unsigned short)cur_rda->seq_no >> 8;
        tmp_value=rwp_table[cur_rwp];
        if (cur_rwp==2)
            cur_rwp=0;
        else
            cur_rwp++;

        outpw(regbase + rwp, tmp_value);

        tmp_value=inpw(regbase + isr);
        if (tmp_value & ISR_RBE)
            outpw(regbase + isr, ISR_RBE);
    }

    /* check rde */
    if (inpw(regbase+isr) & ISR_RDE) {
        outpw(regbase+isr, ISR_RDE);
        tmp_value=inpw(regbase+crda) & 0x0fffe;
        outpw(regbase+crda, tmp_value);
    }

    if (currda == RDANUM-1) {
        currda=0;
        c_rda=rda_start_addr;
        cur_rda=(rda_struct*)c_rda;
    }
    else {
        currda++;
        c_rda+=sizeof(rda_struct);
        cur_rda=(rda_struct*)c_rda;
    }
}

/*
 * init_drv()
 *
 * Initialize the driver and hardware.
 */
init_drv(argc, argv)
int argc;
char *argv[];
{
    char far *ptr;
    int kill_drv;

    fprintf(stderr,
        "%s -- Version %s\n%s\n", drv_name, drv_rev, cpy_msg);

```

TL/F/11720-13



```

kill_drv = do_args(argc, argv);          /* process command line */
sys_isr = _dos_getvect(packet_int_no);   /* get system isr */
ptr = (char far *)sys_isr + 3;

if(kill_drv)                             /* terminate active driver */
    kill_driver(ptr);

if((ptr != NULL) && (far_strcmp(ptr, pkt_signature) == 0)) {
    fprintf(stderr,
        "Error: a packet driver already exist at interrupt 0x%x\n",
        packet_int_no);
    exit(1);
}

_dos_setvect(packet_int_no, drv_isr);     /* install driver isr */

init();          /* init SONIC */

fprintf(stderr,
    "Packet Driver is using INT 0x%x and %ld bytes of memory\n",
    packet_int_no, (unsigned long)mem_sz * 16);
}

/*
 * chk_handle()
 *
 * Check if handle is valid.
 *
 * Return values:  0 - Success
 *                >0 - Failure
 */
chk_handle(handle)
unsigned int handle;
{
    /* check if handle is in range */
    if((handle < MIN_HANDLE) || (handle >= MAX_HANDLES))
        return BAD_HANDLE;

    /* check if handle is in use */
    if(handle_tbl[handle].in_use == 0)
        return BAD_HANDLE;

    return 0;
}

/*
 * kill_driver()
 *
 * Terminate driver from memory
 *
 * Return values:  none - exits from program
 */
kill_driver(ptr)
char far *ptr;
{
    if((ptr == NULL) || (far_strcmp(ptr, pkt_signature) != 0)) {
        fprintf(stderr,
            "Error: no packet driver at interrupt 0x%x\n",

```

TL/F/11720-14

```

        packet_int_no);
    exit(1);
}
r_regs.h.ah = 5;
r_regs.x.bx = 0;
int86(packet_int_no, &r_regs, &r_regs);
if(r_regs.x.cflag) {
    fprintf(stderr, "Error: packet driver can not terminate\n");
    exit(1);
}
printf("Terminated packet driver at interrupt 0x%x\n", packet_int_no);
exit(0);
}

```

```

/*
 * do_args()
 *
 * Process program arguments using getopt().
 *
 * Return values: 0 - Success
 *                1 - Terminate driver
 */
do_args(argc, argv)
int argc;
char *argv[];
{
    int in, done = 0, c_type;
    char *sptr;

    if(argc == 1) /* use default packet_int_no */
        return 0;

#ifdef MSDOS
    if((sptr = strrchr(*argv, '\\')) != NULL)
        strcpy(*argv, sptr + 1);
    if((sptr = strrchr(*argv, '.')) != NULL)
        *sptr = '\0';
#endif

    while (((in = getopt(argc, argv, "?khi:t:")) != -1)) {
        switch(in) {
            case 'k':
                return (1);
                break;
            case 't':
                sscanf(optarg, "%d", &c_type);
                if(c_type==1) cable_type=THICK;
                break;
            case 'i':
                if(sscanf(optarg, "0x%x", &packet_int_no) != 1)
                    if(sscanf(optarg, "%d", &packet_int_no) != 1) {
                        break;
                    }
                /*
                if(!strncmp(optarg, "0x", 2))
                    sscanf(&optarg[2], "%x", &packet_int_no);
                else
                    sscanf(optarg, "%d", &packet_int_no);
                */
        }
    }
}

```

TL/F/11720-15

```

    */
    if((packet_int_no < 0x60) || (packet_int_no > 0x80)) {
        fprintf(stderr,
            "Error: packet_int_no should be in the range 0x60 to 0x80\n");
        exit(1);
    }
    break;
default:
    usage(argv);
    break;
}
}

void usage(argv)
char **argv;
{
    fprintf(stderr,
        "Usage: %s [-h] [-k] [-i packet_int_no] [-t cable type]\n", *argv);
    fprintf(stderr, " -h = this help message\n");
    fprintf(stderr,
        " -i = set packet interrupt number, default is 0x60\n");
    fprintf(stderr, " -t = cable type (0 thin coax, 1 AUI)\n");
    fprintf(stderr, " -k = terminate packet driver\n");
    exit(1);
}

int opterr = 1;
int optind = 1;
char *optarg;
/*
 * getopt() -- Gets options from command line and breaks them up for analysis.
 *             It is functionally compatible with the UNIX version.
 * By Ted Thi
 */
getopt(argc, argv, ctrlStr)
int argc;
char **argv,
    *ctrlStr;
{
    extern char *strchr();
    register char *s_ptr;
    static int i;
    if (optind < argc && argv[optind][++i] == '\0') {
        if (i == 1 || ++optind >= argc)
            return(-1);
        i = 1;
    }
    if (i <= 1) {
        if (optind >= argc || (*argv[optind] != '-' && *argv[optind] != '/') ||
            argv[optind][1] == '\0')
            return(-1);
        if (strcmp(argv[optind] + 1, "-") == 0) {
            optind++;
            return(-1);
        }
    }
    if (argv[optind][i] == ':' || (s_ptr = strchr(ctrlStr, argv[optind][i]))
        == NULL) {
        if (opterr)

```

TL/F/11720-16

```

    fprintf(stderr, "%s: illegal option -- %c\n", *argv, argv[optind][i]);
    return('?');
}
if (s_ptr[1] == ':') {
    if (argv[optind][++i] == '\0') {
        i = 0;
        if (++optind >= argc) {
            if (opterr)
                fprintf(stderr, "%s: option requires an argument -- %c\n", *argv,
                    *s_ptr);
            return('?');
        }
    }
    optarg = argv[optind++] + i;
    i = 0;
} else
    optarg = NULL;
return(*s_ptr);
}
/* of getopt() */

void read_config()
{
    unsigned short reg0, i;
    unsigned short port;

    for(i=0; i<MAX_SLOT; i++) /* read board ID */
    {
        port=(0x1000)*i + ID_ADDR;
        if(inpw(port)==*(unsigned int *)BOARD_ID &&
            inpw(port+2)==*(unsigned int *)BOARD_ID+2)
            break;
    }
    if( i==MAX_SLOT ) { /* no board found */
        fprintf(stderr, "No PLX board found.\n");
        exit (1);
    }

    regbase=0x1000 * i;

    reg0=inp(regbase+0xc88); /* read plx register 0 */
    reg0 &=0x05; /* bit 2,1 */

    switch (reg0) {
        case 0: sonic_irq=5;
                break;
        case 2: sonic_irq=9;
                break;
        case 4: sonic_irq=10;
                break;
        case 6: sonic_irq=11;
                break;
    }

    reg0=inp(regbase+0xc89); /* read plx register 1 */
    if( reg0 & 0x02 ) cable_type=THIN;
    else cable_type=THICK;
}

```

TL/F/11720-17

## FAR.C

```

static char far_rcsid[]="@(#)SID:$";
/*
*****
*           Copyright (c) 1992 National Semiconductor Corporation           *
*           All Rights Reserved   *
*****
*/
#include <dos.h>

void far_memcpy(dest, src, cnt)
register char far *dest;
register char far *src;
register unsigned cnt;
{
    while (cnt--) *dest++ = *src++;
}

char far *far_strcpy(s1, s2)
register char far *s1, far *s2;
{
    char far *s3 = s1;
    while (*s2) *s1++ = *s2++;
    return (s3);
}

far_strcmp(s1, s2)
register char far *s1, far *s2;
{
    while(*s1) {
        if(*s1 != *s2) return(*s1 - *s2);
        s1++; s2++;
    }
    return(*s1 - *s2);
}

far_memcmp(s1, s2, cnt)
register char far *s1, far *s2;
register int cnt;
{
    while(--cnt > 0) {
        if(*s1 != *s2)
            return(*s1 - *s2);
        s1++; s2++;
    }
    return(*s1 - *s2);
}

```

TL/F/11720-18

## ISR.C

```

static char isr_csid[]="@(#) $ID:$";
/*
*****
*           Copyright (c) 1992 National Semiconductor Corporation
*           All Rights Reserved
*****
*/

#include <dos.h>
#include "sonic.h"

#define ISR_STACK_SZ    2048

static char irq_map[] = {
    0x08, 0x09, 0x0a, 0x0b, 0x0c, 0x0d, 0x0e, 0x0f,
    0x70, 0x71, 0x72, 0x73, 0x74, 0x75, 0x76, 0x77
};

static int pic_ctl;
static int pic_mask;
static int old_mask_val;

void (interrupt far *sys_irq_int)();

void interrupt far sonic_isr();

void sonic_isr_enable(irq)
int irq;
{
    pic_ctl = irq < 8 ? 0x20 : 0xa0;
    pic_mask = pic_ctl + 1;

    old_mask_val = inp(pic_mask);
    sys_irq_int = _dos_getVect(irq_map[irq]);

    _disable();
    _dos_setvect(irq_map[irq], sonic_isr);
    outp(pic_mask, old_mask_val & ~(1 << irq%8));
    _enable();

    if(irq>8) { /* also enable PIC 1 */
        int tmp_mask;
        int tmp_pic_ctl;
        int tmp_pic_mask;

        tmp_pic_ctl=0x20;
        tmp_pic_mask = tmp_pic_ctl +1;
        tmp_mask=inp(tmp_pic_mask);
        _disable();
        outp(tmp_pic_mask,tmp_mask & ~(1 << 2));
        _enable();
    }
}

void sonic_isr_disable(irq)
int irq;
{
    _disable();
    _dos_setvect(irq_map[irq], sys_irq_int);
}

```

TL/F/11720-19

```

    outp(pic_mask, old_mask_val);
    _enable();
}

static char far *old_sp;
static char isr_stack[ISR_STACK_SZ];

void interrupt far sonic_isr()
{
    char far *(far get_sp)();
    void (far set_sp)();
    unsigned short activetda, addr;
    unsigned short isr_reg;
    short i;
    tda_struct * tmp_tda;

    outpw(regbase+imr, 0);          /* unmask the imr */

    old_sp = get_sp();
    set_sp((char far *)isr_stack + ISR_STACK_SZ);

    _enable();

    isr_reg=inpw(regbase+isr);

    while (isr_reg) {
        if (isr_reg & ISR_PKTRX) { /* is there a receive */
            outpw(regbase+isr, ISR_PKTRX); /* clear receive bit */
            drv_rcvr(); /* process rda */
        }
        if (isr_reg & ISR_TXDN) { /* is there is transmit done */
            outpw(regbase+isr, ISR_TXDN);
            transmitactive=0;
            for (i=tda_head; i<tda_tail; i++) {
                addr=tda_addr+i*sizeof(tda_struct);
                tmp_tda=(tda_struct *) addr;
                if ((unsigned short)tmp_tda->type==HIGH_PERFORMANCE)
                    xmt_upcall(0, (char far *) &tmp_tda->buffer,
                                (unsigned short)tmp_tda->xmt_dl, (unsigned short)tmp_tda->
            }
        }
        if (isr_reg & ISR_TXER) { /* is there a transmit error */
            outpw(regbase+isr, ISR_TXER);
            if (retry > 10) { /* if retry 10 and still not succeed to transmi
                activetda=inpw(regbase+ctda);
                if (activetda & 0x1)
                    transmitactive=0;
                else {
                    activetda &= 0x0fffe;
                    outpw(regbase+ctda, activetda+20);
                    outp(regbase+cr, 2); /* transmit */
                }
            }
            else { /* try again */
                retry++;
                outp(regbase+cr, 2); /* transmit */
            }
        }
        if (isr_reg & 0x0020)

```

TL/F/11720-20

```
        drv_rcvr();                               /* process rda */
        isr_reg=inpw(regbase+isr);
        isr_reg &=0x0700;
    }

    _disable();
    set_sp(old_sp);
    if(pic_ctl== 0xa0) outp(0x20,0x20);
    outp(pic_ctl, 0x20);
    outpw(regbase+imr, 0x0700);
}
```

TL/F/11720-21



## SONIC.C

```

static char sonic_rcsid[]="@(#)SID:$";
/*
*****
*      Copyright (c) 1992 by National Semiconductor Corporation      *
*              All Rights Reserved                                  *
*****
*/

#include "sonic.h"
#include "dos.h"

/*
 * init()
 *
 * This routine is from init_drv() to initialize sonic buffer and sonic
 * registers.
 *
 * Return values: 0 if success
 *                1 if fail
 */

init()
{
    short i;
    unsigned short cur_loc;

    /* initialize valuables */
    transmitactive=0;
    curtda=0;
    currda=0;

    /* initialize the EISA9010 chip */

        /* register 1 */
    /*cable_type=THICK;*/
    outpw(regbase+plx_reg1,cable_type);

    /* install sonic interrupt */
    sonic_isr_enable(sonic_irq);

    /* initialize sonic register */
    outpw(regbase+cr, 0x94); /* reset sonic */
    outpw(regbase+dcr, 0x073a); /* set configuration: 0 wait state
                                32-bit data path
                                block mode
                                8 words receive fifo
                                12 words transmit fifo */

    outpw(regbase+cr, 0); /* out of reset mode */
    outpw(regbase+rcr, 0x2000); /* accept broadcast packet */
    outpw(regbase+isr, 0x0fff); /* reset isr */
    outpw(regbase+imr, 0x0700); /* set mask to xmit done, xmit error and
                                receive packet */

    init_tda(); /* init tda */
    init_rda(); /* init rda */
    init_rra(); /* init rra */
    init_cam(); /* init cam */
}

```

TL/F/11720-22

```

/* initialize rwp location table */
cur_loc=inpw(regbase+rsa);
for (i=0; i<RRANUM; i++) {
    rwp_table[i]=cur_loc;
    cur_loc+=16;
}
cur_rwp=0;

/* normal operation */
outpw(regbase+cr, 0x100); /* read rra */

return(0);
}

/*
 * init_tda()
 *
 * This routine is to link the tda so as to make transmission more
 * efficient. It also initialize the utda and ctda registers.
 */
init_tda()
{
    unsigned short i, u16, l16;
    unsigned long addr32;
    unsigned long tba_addr;
    char far *ptr;
    struct SREGS segregs;
    tda_struct *tmp_tda;
    unsigned short c_tda_addr;
    unsigned short n_tda_addr;

    segread(&segregs); /* Read the segment register value */
    /* check double word boundary */
    tda_addr=(unsigned short) &tda[0];
    tda_addr&=0xffff;
    /* link the first nine tda */
    for (i=0; i<TDANUM-1; i++) {
        c_tda_addr=tda_addr+i*sizeof(tda_struct);
        n_tda_addr=c_tda_addr+sizeof(tda_struct);
        addr32=((unsigned long) segregs.ds << 16) | n_tda_addr;
        tba_addr=((unsigned long)segregs.ds << 16) |
            ((unsigned short) &tba[i]);
        u16=addr32>>16;
        l16=(unsigned short)addr32;
        addr32=(unsigned long)u16 * 16 + l16;
        tmp_tda=(tda_struct*) c_tda_addr;
        tmp_tda->config=0x1000;
        tmp_tda->link=(unsigned short) addr32;
        u16=tba_addr>>16;
        l16=(unsigned short)tba_addr;
        tba_addr=(unsigned long)u16 * 16 + l16;
        tmp_tda->frag_ptr1=tba_addr>>16;
        tmp_tda->frag_ptr0=(unsigned short) tba_addr;
    }
}

```

TL/F/11720-23

```

/* set the last tda link field to the first tda */
addr32=((unsigned long) segregs.ds << 16) | tda_addr);
tba_addr=((unsigned long) segregs.ds << 16) |
          ((unsigned short) &tba[TDANUM-1]));
u16=addr32>>16;
l16=(unsigned short)addr32;
addr32=(unsigned long)u16 * 16 + l16;
c_tda_addr=tda_addr+(TDANUM-1)*sizeof(tda_struct);
tmp_tda=(tda_struct*) c_tda_addr;
tmp_tda->link=(unsigned short) addr32;
u16=tba_addr>>16;
l16=(unsigned short)tba_addr;
tba_addr=(unsigned long)u16 * 16 + l16;
tmp_tda->frag_ptr1=tba_addr>>16;
tmp_tda->frag_ptr0=(unsigned short) tba_addr;

/* set the utda and ctda register */
outpw(regbase+utda, addr32>>16); /* set utda */
outpw(regbase+ctda, (unsigned short)addr32); /* set ctda */
tda_start_addr=(unsigned short)addr32;
}

```

```

/*
 * init_rda()
 *
 * This routine is to link the rda together. It also initialize the urda and
 * crda registers.
 */

```

```

init_rda()
{
  unsigned short i, u16, l16;
  unsigned long addr32;
  struct SREGS segregs;
  rda_struct *tmp_rda;
  unsigned short c_rda_addr;
  unsigned short n_rda_addr;

  segread(&segregs); /* Read the segment register value */

  /* check double word boundry */
  rda_addr=(unsigned short) &rda[0];
  rda_addr&=0xffffc;
  c_rda=rda_addr;
  rda_start_addr=c_rda;
  cur_rda=(rda_struct *) c_rda;
  /* Link the Rda */
  for (i=0; i<RDANUM-1; i++) {
    c_rda_addr=rda_addr+i*sizeof(rda_struct);
    n_rda_addr=c_rda_addr+sizeof(rda_struct);
    addr32=((unsigned long) segregs.ds << 16) | n_rda_addr);
    u16=addr32>>16;
    l16=(unsigned short)addr32;
    addr32=(unsigned long)u16 * 16 + l16;
  }
}

```

```

    tmp_rda=(rda_struct*) c_rda_addr;
    tmp_rda->pkt_link=(unsigned short) addr32;
    tmp_rda->in_use=0x0ffff;
}

/* set the last rda link field to the first rda */
addr32=((unsigned long) segregs.ds << 16) | rda_addr);
u16=addr32>>16;
l16=(unsigned short)addr32;
addr32=(unsigned long)u16 * 16 + l16;
c_rda_addr=rda_addr+(RDANUM-1)*sizeof(rda_struct);
tmp_rda=(rda_struct*) c_rda_addr;
tmp_rda->in_use=0x0ffff;
tmp_rda->pkt_link=(unsigned short) addr32;
tmp_rda->pkt_link|=1; /* set EOL */

/* set the urda and crda register */
outpw(regbase+urda, addr32>>16); /* set urda */
outpw(regbase+crda, (unsigned short)addr32); /* set crda */
}

/*
 * init_rra()
 *
 * This routine is initialize the rra and set rsa, rea, rrp, rwp registers
 */
init_rra()
{
    unsigned short i, u16, l16;
    unsigned long addr32;
    struct SREGS segregs;
    unsigned short rra_addr, addr;
    rra_struct * tmp_rra;

    segread(&segregs); /* Read the segment register value */

    /* check double word boundry */
    rra_addr=(unsigned short) &rra[0];
    rra_addr&=0xffffc;
    /* Initialize the rra slot */
    for (i=0; i<RRANUM; i++) {
        addr32=((unsigned long) segregs.ds << 16) |
            ((unsigned short) &rba[i]));
        u16=addr32>>16;
        l16=(unsigned short)addr32;
        addr32=(unsigned long)u16 * 16 + l16;
        addr=rra_addr+i*sizeof(rra_struct);
        tmp_rra=(rra_struct*) addr;
        tmp_rra->buff_ptr0=(unsigned short)addr32;
        tmp_rra->buff_ptr1=addr32>>16;
        tmp_rra->buff_wc0=RBA_BUF_SIZE/2;
        tmp_rra->buff_wc1=0;
    }

    addr32=((unsigned long) segregs.ds << 16) | rra_addr);

```

```

u16=addr32>>16;
l16=(unsigned short)addr32;
addr32=(unsigned long)u16 * 16 + l16;

/* set urra, rsa, and rrp */
outpw(regbase+urra, addr32 >> 16); /* set urra */
outpw(regbase+rsa, (unsigned short)addr32); /* set rsa */
outpw(regbase+rrp, (unsigned short)addr32); /* set rrp */

/* set rea and rwp */
addr32+=48;
outpw(regbase+rea, (unsigned short) addr32); /* set rea */
outpw(regbase+rwp, (unsigned short) addr32); /* set rwp */

}

/*
 * init_cam()
 *
 * This routine is initialize the cam and set cdp, cdc registers. Also,
 * load the cam.
 *
 */
init_cam()
{
    unsigned short i, u16, l16;
    unsigned long addr32;
    struct SREGS segregs;
    unsigned short cam_addr, addr;
    cam_struct * tmp_cam;

    segread(&segregs); /* Read the segment register value */
    /* check double word boundary */
    cam_addr=(unsigned short) &cam[0];
    cam_addr&=0xffffc;

    addr32=((unsigned long) segregs.ds << 16) | cam_addr;
    u16=addr32>>16;
    l16=(unsigned short)addr32;
    addr32=(unsigned long)u16 * 16 + l16;

    outpw(regbase+cdp, (unsigned short) addr32); /* load cdp */
    outpw(regbase+cdc, 16); /* load cdc */

    tmp_cam=(cam_struct *) cam_addr;
    /* load the cda with node physical address */
    tmp_cam->cam_port_info[0].port0=inpw(regbase+0x0c90);
    tmp_cam->cam_port_info[0].port1=inpw(regbase+0x0c92);
    tmp_cam->cam_port_info[0].port2=inpw(regbase+0x0c94);

    for(i=0; i<16; i++)
        tmp_cam->cam_port_info[i].entry_ptr=i;

    tmp_cam->cam_enable=1; /* load cam enable */

    /* load cam */
    outpw(regbase+cr, CMD_LCAM);

    /* to ensure load cam is properly executed and clear LCD bit in isr */
    for (;;) {
        if (inpw(regbase+isr) & ISR_LCD) {
            outpw(regbase+isr, ISR_LCD);
            break;
        }
    }
}

```

TL/F/11720-26

TL/F/11720-27

## PKTDRV.H

```

/*
 * $ID:$
 *
 ****
 *          Copyright (c) 1990 by National Semiconductor Corporation      *
 *                    All Rights Reserved                                *
 ****
 */

/* Packet Driver Error numbers */
#define BAD_HANDLE      1      /* invalid handle number */
#define NO_CLAS        2      /* no interfaces of specified class found */
#define NO_TYPE        3      /* no interfaces of specified type found */
#define NO_NUMBER      4      /* no interfaces of specified number found */
#define BAD_TYPE       5      /* bad packet type specified */
#define NO_MULTICAST   6      /* this interface does not support multicast */
#define CANT_TERMINATE 7      /* this packet driver cannot terminate */
#define BAD_MODE       8      /* an invalid receiver mode was specified */
#define NO_SPACE       9      /* failed because of insufficient space */
#define TYPE_INUSE     10     /* the type has already been accessed */
                                /* and not released. */
#define BAD_COMMAND    11     /* command out of range, or not implemented */
#define CANT_SEND      12     /* packet couldn't be sent (usually hardware) */
#define CANT_SET       13     /* hardware address couldn't be changed */
                                /* (more than 1 handle open) */
#define BAD_ADDRESS    14     /* hardware address has bad length or format */
#define CANT_RESET     15     /* couldn't reset interface */
                                /* (more than 1 handle open) */

#define RUNT           60     /* smallest legal size packet, no fcs */
#define GIANT          1514   /* largest legal size packet, no fcs */
#define EADDR_LEN      6      /* Ethernet address length. */

#define MAX_HANDLES 10      /* max number of handles at one time */
#define MIN_HANDLE 0       /* handles are 0 thru 9 */
#define MAX_TYPE_LEN 2     /* max packet type length */
#define OPEN -1           /* available handle */

#define MIN(a,b) ((a) < (b)) ? (a) : (b)
#define MAX(a,b) ((a) > (b)) ? (a) : (b)

/* handle structure */
typedef struct _handle {
    int in_use;          /* non-zero if handle exist */
    char type[MAX_TYPE_LEN]; /* packet type */
    int len;            /* packet length */
    unsigned int rec_es; /* receiver address segment */
    unsigned int rec_di; /* receiver address offset */
} HANDLE;

static unsigned char bit_swap[256] = {
    0x00, 0x80, 0x40, 0xc0, 0x20, 0xa0, 0x60, 0xe0,
    0x10, 0x90, 0x50, 0xd0, 0x30, 0xb0, 0x70, 0xf0,
    0x08, 0x88, 0x48, 0xc8, 0x28, 0xa8, 0x68, 0xe8,
    0x18, 0x98, 0x58, 0xd8, 0x38, 0xb8, 0x78, 0xf8,
    0x04, 0x84, 0x44, 0xc4, 0x24, 0xa4, 0x64, 0xe4,
    0x14, 0x94, 0x54, 0xd4, 0x34, 0xb4, 0x74, 0xf4,
    0x0c, 0x8c, 0x4c, 0xcc, 0x2c, 0xac, 0x6c, 0xec,

```

TL/F/11720-28

```

0x1c, 0x9c, 0x5c, 0xdc, 0x3c, 0xbc, 0x7c, 0xfc,
0x02, 0x82, 0x42, 0xc2, 0x22, 0xa2, 0x62, 0xe2,
0x12, 0x92, 0x52, 0xd2, 0x32, 0xb2, 0x72, 0xf2,
0x0a, 0x8a, 0x4a, 0xca, 0x2a, 0xaa, 0x6a, 0xea,
0x1a, 0x9a, 0x5a, 0xda, 0x3a, 0xba, 0x7a, 0xfa,
0x06, 0x86, 0x46, 0xc6, 0x26, 0xa6, 0x66, 0xe6,
0x16, 0x96, 0x56, 0xd6, 0x36, 0xb6, 0x76, 0xf6,
0x0e, 0x8e, 0x4e, 0xce, 0x2e, 0xae, 0x6e, 0xee,
0x1e, 0x9e, 0x5e, 0xde, 0x3e, 0xbe, 0x7e, 0xfe,
0x01, 0x81, 0x41, 0xc1, 0x21, 0xa1, 0x61, 0xe1,
0x11, 0x91, 0x51, 0xd1, 0x31, 0xb1, 0x71, 0xf1,
0x09, 0x89, 0x49, 0xc9, 0x29, 0xa9, 0x69, 0xe9,
0x19, 0x99, 0x59, 0xd9, 0x39, 0xb9, 0x79, 0xf9,
0x05, 0x85, 0x45, 0xc5, 0x25, 0xa5, 0x65, 0xe5,
0x15, 0x95, 0x55, 0xd5, 0x35, 0xb5, 0x75, 0xf5,
0x0d, 0x8d, 0x4d, 0xcd, 0x2d, 0xad, 0x6d, 0xed,
0x1d, 0x9d, 0x5d, 0xdd, 0x3d, 0xbd, 0x7d, 0xfd,
0x03, 0x83, 0x43, 0xc3, 0x23, 0xa3, 0x63, 0xe3,
0x13, 0x93, 0x53, 0xd3, 0x33, 0xb3, 0x73, 0xf3,
0x0b, 0x8b, 0x4b, 0xcb, 0x2b, 0xab, 0x6b, 0xeb,
0x1b, 0x9b, 0x5b, 0xdb, 0x3b, 0xbb, 0x7b, 0xfb,
0x07, 0x87, 0x47, 0xc7, 0x27, 0xa7, 0x67, 0xe7,
0x17, 0x97, 0x57, 0xd7, 0x37, 0xb7, 0x77, 0xf7,
0x0f, 0x8f, 0x4f, 0xcf, 0x2f, 0xaf, 0x6f, 0xef,
0x1f, 0x9f, 0x5f, 0xdf, 0x3f, 0xbf, 0x7f, 0xff,
};
#define BIT_SWAP(a) bit_swap[(unsigned char )(a)]

#define BYTE_SWAP(a, b) { *(a) = *(b+1); *(a+1) = *(b); }

#define BUF_SZ 1514
static unsigned char s_buf[BUF_SZ];

static unsigned char snap[] =
/* SNAP */
{ 170, 170, 3, 0, 0, 0 };

#define ETYPE_OFS 12
#define DATA_OFS 14
#define MAC_LEN 14

static struct {
    unsigned long packets_in;
    unsigned long packets_out;
    unsigned long bytes_in;
    unsigned long bytes_out;
    unsigned long errors_in;
    unsigned long errors_out;
    unsigned long packets_dropped;
} drv_stats;

static struct {
    unsigned char major_rev;
    unsigned char minor_rev;
    unsigned char length;
    unsigned char addr_len;
    unsigned short mtu;
    unsigned short multicast_aval;
    unsigned short rcv_bufs;
    unsigned short xmt_bufs;

    unsigned short int_num;
} drv_param;

```

TL/F/11720-29

TL/F/11720-30

## SONIC.H

```

/*
 * $ID:$
 *
 *****
 * Copyright (c) 1990 by National Semiconductor Corporation *
 * All Rights Reserved *
 *****
 */

/* SONIC definition and data structures */

#define TDANUM 5
#define RDANUM 40
#define RRANUM 3
#define RBA_BUF_SIZE 8192
#define TBA_BUF_SIZE 1514

/* isr bit pattern */
#define CMD_LCAM 0x0200
#define ISR_RFO 0x0001
#define ISR_RBE 0x0020
#define ISR_RDE 0x0040
#define ISR_PKTRX 0x0400
#define ISR_TXDN 0x0200
#define ISR_TXER 0x0100
#define ISR_LCD 0x1000

#define THIN 0x03
#define THICK 0x01
#define ID_ADDR 0xC80
#define MAX_SLOT 15

/*****
 *
 * Offset of the EISA9010 register from the regbase address *
 *
 *****/
#define plx_ebc 0xC84 /* EBC register */
#define plx_reg0 0xC88 /* register 0 */
#define plx_reg1 0xC89 /* register 1 */
#define plx_reg2 0xC8A /* register 2 */
#define plx_reg3 0xC8F /* register 3 */
/*****
 *
 * Offset of the register from the i/o base address *
 *
 *****/

#define cr 0 /* Command */
#define dcr 2 /* Data Configuration */
#define rcr 4 /* Receive Control */
#define tcr 6 /* Transmit Control */
#define imr 8 /* Interrupt Mask */
#define isr 10 /* Interrupt Status */
#define utda 12 /* Upper Transmit Descriptor Addr */
#define ctda 14 /* Current Transmit Descriptor Addr */
#define tps 16 /* Transmit Packet Size */
#define tfc 18 /* Transmit Fragment Count */
#define tsa0 20 /* Transmit Start Address 0 */

```

TL/F/11720-31



```

#define tsal      22 /* Transmit Start Address 1 */
#define tfs      24 /* Transmit Fragment Size */
#define urda     26 /* Upper Receive Descriptor Addr */
#define crda     28 /* Current Receive Descriptor Addr */
#define crba0    30 /* Current Receive Buffer Addr 0 */
#define crba1    32 /* Current Receive Buffer Addr 1 */
#define rbwc0    34 /* Remaining Buffer Word Count 0 */
#define rbwc1    36 /* Remaining Buffer Word Count 1 */
#define eobc     38 /* End of Buffer Word Count */
#define urra     40 /* Upper Receive Resource Addr */
#define rsa      42 /* Resource Start Addr */
#define rea      44 /* Resource End Addr */
#define rrp      46 /* Resource Read Addr */
#define rwp      48 /* Resource Write Addr */
#define trba0    50 /* Temp Recv. Buffer Addr 0 */
#define trba1    52 /* Temp Recv. Buffer Addr 1 */
#define tbwc0    54 /* Temp Buffer Word Count 0 */
#define tbwc1    56 /* Temp Buffer Word Count 1 */
#define addr0    58 /* Address Generator 0 */
#define addr1    60 /* Address Generator 1 */
#define llfa     62 /* Last link Field Addr */
#define ttda     64 /* Temp Transmit Descriptor Addr */
#define cep      66 /* CAM entry Point */
#define cap2     68 /* CAM Address Port 2 */
#define cap1     70 /* CAM Address Port 1 */
#define cap0     72 /* CAM Address Port 0 */
#define ce       74 /* CAM Enable */
#define cdp      76 /* CAM Descriptor Pointer */
#define cdc      78 /* CAM Descriptor Count */
#define sr       80 /* Silicon Revision */
#define wt0      82 /* Watchdog Timer 0 */
#define wt1      84 /* Watchdog Timer 1 */
#define rsc      86 /* Receive Sequence Counter */
#define crct     88 /* CRC Error Tally */
#define faet     90 /* FAE Error Tally */
#define mpt      92 /* Missed Packet Tally */
#define mdt      94 /* Maximum Deferral Timer */
#define rtc      96 /* Receive Test Control */
#define ttc      98 /* Transmit Test Control */
#define dtc     100 /* DMA Test Control */
#define cc0     102 /* CAM Comparison 0 */
#define cc1     104 /* CAM Comparison 1 */
#define cc2     106 /* CAM Comparison 2 */
#define cm      108 /* CAM Match */
#define reserve1 110 /* Reserved */
#define reserve2 112 /* Reserved */
#define rbc     114 /* Receiver Byte Count */
#define reserve3 116 /* Reserved */
#define tbc     118 /* Transmitter Backoff Counter */
#define trc     120 /* Transmitter Random Counter */
#define tbm     124 /* Transmitter Backoff Mask */
#define reserve4 126 /* Reserved */
#define reserve5 128 /* Reserved */

#define BASIC      0
#define HIGH_PERFORMANCE 1

/* tda structure */
typedef struct tda_construct {
    unsigned long    status;

```

TL/F/11720-32

```

    unsigned long   config;
    unsigned long   pkt_size;
    unsigned long   frag_count;
    unsigned long   frag_ptr0;
    unsigned long   frag_ptr1;
    unsigned long   frag_size;
    unsigned long   link;
    unsigned long   type;
    char far *      buffer;
    unsigned long   xmt_di;
    unsigned long   xmt_es;
}   tda_struct;

/* rda structure */
typedef struct rda_construct {
    unsigned long   status;
    unsigned long   byte_count;
    unsigned long   pkt_ptr0;
    unsigned long   pkt_ptr1;
    unsigned long   seq_no;
    unsigned long   pkt_link;
    unsigned long   in_use;
}   rda_struct;

/* rra structure */
typedef struct rra_construct {
    unsigned long   buff_ptr0;
    unsigned long   buff_ptr1;
    unsigned long   buff_wc0;
    unsigned long   buff_wcl;
}   rra_struct;

/* rba structure */
typedef struct rba_construct {
    unsigned char   buff[RBA_BUF_SIZE];
}   rba_struct;

/* tba structure */
typedef struct tba_construct {
    unsigned char   tba_buff[TBA_BUF_SIZE];
}   tba_struct;

typedef struct cam_port {
    unsigned long   entry_ptr;
    unsigned long   port0;
    unsigned long   port1;
    unsigned long   port2;
}   cam_port_struct;

typedef struct cam_construct {
    cam_port_struct cam_port_info[16];
    unsigned long   cam_enable;
}   cam_struct;

rba_struct   rba[RRANUM];
tba_struct   tba[TDANUM];
unsigned char tda[TDANUM*sizeof(tda_struct)+3];
unsigned char rda[RDANUM*sizeof(rda_struct)+3];
unsigned short in_isr;

```

TL/F/11720-33

```
unsigned char rra[RRANUM*sizeof(rda_struct)+3];
unsigned char cam[sizeof(cam_struct)+3];

unsigned short sonic_irq;          /* sonic interrupt*/
unsigned short cable_type;        /* thin/thick cable */
unsigned short regbase;           /* base io address */
short transmitactive;             /* transmission currently active flag */
short curtda;                     /* current tda */
short currda;                     /* current rda */
short previous_seqno;             /* previous sequence number */
short retry;                       /* transmit retry counter */
unsigned short rwp_table[6];      /* RRA location table structure */
short cur_rwp;                    /* pointer to rwp_table */
unsigned short tda_addr;          /* tda starting address */
unsigned short tda_start_addr;    /* tda starting physical address */
unsigned short rda_addr;         /* rda starting address */
unsigned short c_rda;
unsigned short rda_start_addr;
unsigned char far *type_ptr;      /* pointer for packet type */
short tda_head;                  /* head ptr to tda list */
short tda_tail;                  /* tail ptr to tda list */
rda_struct * cur_rda;
```

TL/F/11720-34

## ISRLIB.ASM

```

;
;*****
;*          Copyright (c) 1990 National Semiconductor Corporation      *
;*          All Rights Reserved                                       *
;*****

_TEXT SEGMENT WORD PUBLIC 'CODE'
_TEXT ENDS
_DATA SEGMENT WORD PUBLIC 'DATA'
_DATA ENDS
_CONST SEGMENT WORD PUBLIC 'CONST'
_CONST ENDS
_BSS SEGMENT WORD PUBLIC 'BSS'
_BSS ENDS
DGROUP GROUP CONST, _BSS, _DATA
        ASSUME CS: _TEXT, DS: DGROUP, SS: DGROUP

_TEXT segment word public 'CODE'
        assume cs:_TEXT

        public _get_sp
_get_sp proc far
        mov     ax,sp
        add     ax,4
        mov     dx,ss
        ret

_get_sp ENDP

        public _set_sp
_set_sp proc far
        mov     bx,ss
        mov     es,bx
        mov     bx,sp

        pushf
        cli
        pop     dx

        mov     sp,word ptr ss:[bx+4]
        mov     ss,word ptr ss:[bx+6]

        and     dx,512
        jz     skip
        sti

skip:    sub     sp,4
        mov     ax, word ptr es:[bx+2]
        push   ax
        mov     ax, word ptr es:[bx]
        push   ax
        ret

_set_sp ENDP

        public _get_if
_get_if proc far
        pushf
        pop     dx
        mov     ax,0
        and     dx,512

```

TL/F/11720-35

```

        jz      ifret
        mov     ax,1
ifret:  ret
_get_if      ENDP

ARG_OFS equ 6 ;near = 4, far = 6 (from bp)
public _int_fddi
_int_fddi   proc far
        push   bp
        mov    bp, sp
        sub    sp, 8 ;work area for INT code

        ;put INT code on stack
        mov    byte ptr[bp - 2], 0cbh
        mov    ax, word ptr[bp + ARG_OFS]
        mov    [bp - 3], al
        mov    byte ptr[bp - 4], 0cdh
        mov    word ptr[bp - 6], ss
        lea   ax, word ptr[bp - 4]
        mov    word ptr[bp - 8], ax

        ;get regs values off sp, pointers are far
        push   bp
        mov    es, [bp + ARG_OFS + 4]
        mov    bp, [bp + ARG_OFS + 2]
        mov    ax, es:[bp]
        mov    bx, es:[bp + 2]
        mov    cx, es:[bp + 4]
        mov    dx, es:[bp + 6]
        mov    si, es:[bp + 8]
        mov    di, es:[bp + 10]
        pop    bp

        call   dword ptr[bp - 8] ;do INT

        ;get carry bit
        push   ax
        pushf
        pop    ax
        and    ax, 1 ;mask carry bit

        ;put regs values on sp
        mov    es, [bp + ARG_OFS + 8]
        mov    bp, [bp + ARG_OFS + 6]
        mov    es:[bp + 12], ax ;cflag
        pop    ax
        mov    es:[bp], ax
        mov    es:[bp + 2], bx
        mov    es:[bp + 4], cx
        mov    es:[bp + 6], dx
        mov    es:[bp + 8], si
        mov    es:[bp + 10], di

        add    sp, 8
        pop    bp
        ret
_int_fddi      ENDP

_TEXT ends
end

```

## PKTINT.ASM

```

;
; *****
; *          Copyright (c) 1990 by National Semiconductor Corporation          *
; *                      All Rights Reserved                                  *
; *****
;
; title TEXT - Interrupt service routine
;
; extrn _int_handler:near
;
; _TEXT SEGMENT WORD PUBLIC 'CODE'
; _TEXT ENDS
; _DATA SEGMENT WORD PUBLIC 'DATA'
; _DATA ENDS
; CONST SEGMENT WORD PUBLIC 'CONST'
; CONST ENDS
; _BSS SEGMENT WORD PUBLIC 'BSS'
; _BSS ENDS
; DGROUP GROUP CONST, _BSS, _DATA
; ASSUME CS: _TEXT, DS: DGROUP, SS: DGROUP
;
; _DATA SEGMENT WORD PUBLIC 'DATA'
; assume ds:DGROUP
; rcvr_ptr dd ?
; upcall_ptr dd ?
; segmoffs struc
; offs dw ?
; segm dw ?
; segmoffs ends
;
; _DATA ENDS
;
; _TEXT segment word public 'CODE'
; assume cs:_TEXT
;
; CFLAG_OFFSET equ 2
; FLAG_OFFSET equ 6
; REGS_OFFSET equ 14
; SREGS_OFFSET equ 22
;
; public _drv_isr
; _drv_isr proc far
;
; jmp start
; db 'PKT DRVR',0 ;driver signature
;
; setup registers on stack for MSC's union REGS and struct SREGS
; start:
;
; assume ds:nothing
;
; push bp
; mov bp, sp
; and word ptr[bp+FLAG_OFFSET], not 1 ;clear carry bit
; push word ptr[bp+FLAG_OFFSET] ;put in cflag field of structure
; push di ;save regular registers

```

TL/F/11720-37

```

push    si
push    dx
push    cx
push    bx
push    ax
push    ds                ;save segment registers
push    ss
push    cs
push    es

push    ss
lea     ax, word ptr [bp-SREGS_OFFSET] ;pass sregs pointer
push    ax
push    ss
lea     ax, word ptr [bp-REGS_OFFSET]  ;pass regs pointer -> ax
push    ax

mov     ax, DGROUP        ;get global data segment
mov     ds, ax            ;make segment addressable
assume ds: DGROUP
cld
call    _int_handler      ;call C interrupt handler
add     sp, 8

mov     ax, word ptr [bp-CFLAG_OFFSET] ;mov cflag to flag reg
mov     word ptr [bp+FLAG_OFFSET], ax

pop     es                ;restore registers
pop     ax                ;dummy pop for cs
pop     ss
pop     ds
pop     ax
pop     bx
pop     cx
pop     dx
pop     si
pop     di
pop     bp                ;pop cflag of structure
pop     bp

iret                    ;return from interrupt
_drv_isr                endp

public _app_rcv
_app_rcv proc near
ax_ofs equ 4

assume ds:DGROUP
push    bp
mov     bp, sp
push    ds
push    es
push    bx

mov     bx, [bp+ax_ofs+10] ;set-up app reciever
mov     rcvr_ptr.offset, bx
mov     bx, [bp+ax_ofs+12]
mov     rcvr_ptr.segm, bx

```

TL/F/11720-38

```

    les     bx, dword ptr[bp+ax_ofs+6] ;buffer
    mov     si, word ptr es:[bx]
    push   ds
    mov     ds, word ptr es:[bx+2]
    mov     ax, [bp+ax_ofs]
    mov     bx, [bp+ax_ofs+2]
    mov     cx, [bp+ax_ofs+4]
    pop     es
    assume es:DGROUP

    call    es:rcvr_ptr

    mov     ax, es
    les     bx, dword ptr[bp+ax_ofs+6]      ;update pointer ES:DI
    mov     word ptr es:[bx], di
    mov     word ptr es:[bx+2], ax
    pop     bx
    pop     es
    pop     ds
    pop     bp
    ret

_app_rcv     endp

        public _xmt_upcall
_xmt_upcall proc near
ret_ofs equ 4

    assume ds:DGROUP
    push   bp
    mov     bp, sp
    push   ds
    push   es
    push   bx

    mov     bx, [bp+ret_ofs+6]
    mov     upcall_ptr.offs, bx
    mov     bx, [bp+ret_ofs+8]
    mov     upcall_ptr.segm, bx

    les     bx, dword ptr[bp+ret_ofs+2] ;buffer
    mov     di, word ptr ds:[bx]
    mov     es, word ptr ds:[bx+2]
    mov     ax, [bp+ret_ofs]
    assume ds:DGROUP

    call    ds:upcall_ptr

    pop     bx
    pop     es
    pop     ds
    pop     bp
    ret

_xmt_upcall     endp

_TEXT     ends
end

```



## MAKEFILE

```
ZI      = -Zi
INC     = ..\include
CFLAGS = $(ZI) -Gs -I$(INC) -c
MFLAGS = -Ml

OBJ     = pktdrv.obj sonic.obj pktint.obj far.obj isr.obj isrlib.obj
LIB     =

sonic.obj: sonic.c $(INC)\sonic.h
        cl $(CFLAGS) $*.c

pktdrv.obj: pktdrv.c $(INC)\pktdrv.h $(INC)\sonic.h
        cl $(CFLAGS) $*.c

far.obj: far.c $(INC)\sonic.h
        cl $(CFLAGS) $*.c

isr.obj: isr.c $(INC)\sonic.h
        cl $(CFLAGS) $*.c

isrlib.obj: isrlib.asm
        masm $(MFLAGS) $*.asm;

pktint.obj: pktint.asm
        masm $(MFLAGS) $*.asm;

pktdrv.exe: $(OBJ)
        cl $(ZI) $(OBJ) -o $*

clean:
        -del *.obj
```

TL/F/11720-40

# DP83932EB-EISA SONIC™ EISA Bus Master Ethernet Adapter

National Semiconductor  
Application Note 877



## INTRODUCTION

The purpose of this application note is to describe the implementation of an EISA bus master Ethernet interface solution using National Semiconductor's DP83932 System Oriented Network Interface Controller (SONIC™) and PLX Technology's EISA9032 EISA Bus Master Interface chip.

This solution takes the form of a high performance 32-bit network interface adapter card which on one side plugs into an EISA bus slot and on the other supports two media connection options, Attachment Unit Interface (AUI) and Thin wire Ethernet.

The board easily interfaces to the EISA bus with few external components. This application note assumes the reader is familiar with National Semiconductor's DP83932 SONIC™ Ethernet controller, PLX Technology's EISA9032 EISA interface chip and the EISA bus specification.

This document will first give a hardware functional description of the card, followed by an overview of EISA covering topics such as system configuration, I/O access, multiple bus masters and bus protocol, and ending with a description of the master and slave interfaces of the Ethernet board.

## HARDWARE FUNCTIONAL OVERVIEW

The main function of this adapter card is to transfer Ethernet packet data to/from the CPU's system memory as a high speed 32-bit bus master during LAN transmissions and receptions at the maximum EISA burst rate of 33 Mbytes/s.

A 32-bit bus master architecture, in which the SONIC Ethernet controller can gain ownership of the EISA bus and transfer data directly into system memory with no on-board CPU or buffer RAM has been chosen for this design to maximize data throughput while not adding any extra memory cost or intelligence on the card. In addition the inherent packet buffer management features of SONIC are utilized by driver software to facilitate optimum performance. The card has a typical (calculated) bus occupancy of  $\leq 10\%$  for full Ethernet traffic (10 Mb/s).

The block diagram of this board is shown in *Figure 1*. The design can be broken down into 3 sections: slave interface, bus master interface, and physical media interface.

The slave interface enables the EISA host CPU to gain access to the following devices on the adapter card:

1. 32 x 8 PROM which contains the card's Ethernet node ID, and the card's EISA ID.
2. An optional 256k x 8 boot EPROM which can contain a program which enables a diskless CPU to boot up across the network.
3. The SONIC Ethernet controller internal registers.
4. The EISA9032 EISA interface chip configuration registers.

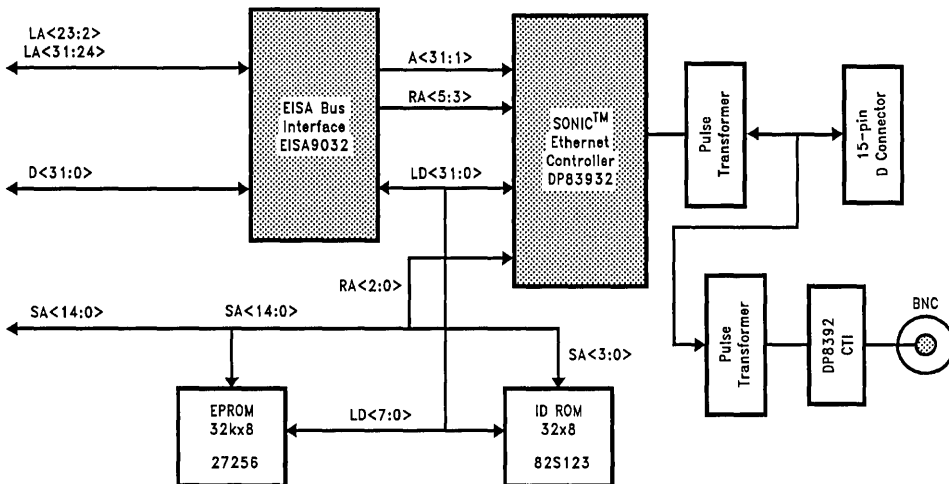


FIGURE 1. SONIC EISA Ethernet Adapter Card Block Diagram

TL/F/11788-1

The master interface enables the SONIC Ethernet controller to read and write to the system memory on the EISA bus using the EISA9032 interface chip. The EISA9032 interface chip converts the Ethernet controller's arbitration and cycle control signals to the EISA bus timing and protocol.

The physical interface enables the SONIC Ethernet controller to transmit and receive data over a 10BASE5 thick wire Ethernet interface or the 10BASE2 thin wire interface using National Semiconductor DP8392 Coaxial Interface Transceiver CTI.

### Transmission

The sequence of events for Ethernet transmissions is as follows:

The host CPU writes the packet data into the system's memory Transmit Buffer Area (TBA). It then writes descriptor information (packet data pointers, packet size, etc.) into the system memory transmit descriptor area (TDA). Next it loads a SONIC register with a pointer to the TDA and issues a transmit command by writing to the SONIC's command register.

The SONIC responds by first reading the TDA descriptor information from system memory. It then loads the packet data from the system memory TBA into its internal FIFO in bursts and transmits this data onto the network. At the end of the transmission the SONIC will write transmit status information into the system's memory TDA.

### Reception

The sequence of events for Ethernet receptions is as follows:

Data is loaded from the Ethernet cable into the SONIC's internal FIFO. When a programmable threshold is reached in the FIFO, the SONIC will write the packet data into the system memory's Receive Buffer Area (RBA).

Once a complete packet has been loaded into memory the SONIC will write descriptor information about the reception into the system memory's Receive Descriptor Area (RDA).

Note that all buffer and descriptor areas are set up by the host CPU in system memory prior to any packet transmission and reception.

### EISA OVERVIEW

EISA was developed in 1989 by a consortium of 9 PC manufacturers in an attempt to create a higher performance 32-bit bus architecture that is backwards compatible with the PC-AT® based industry Standard Architecture (ISA) created in 1984.

This section gives an overview of the Extended Industry Standard Architecture EISA and describes the Ethernet adapter's implementation of its interface. First, the bus features are described, then various facets of bus operation are described, including addressing, arbitration, configuration, and the bus protocol.

### Bus Features

- 64 kBytes of I/O space; Slot specific I/O access
- 32-bit non multiplexed address data bus supporting a 4 GByte address range

- Multiple bus masters using a centralized arbitration scheme supporting preemption
- Synchronous protocol (8.3 MHz clock) supporting standard (2 bus clock per cycle) or burst (1 bus clock per cycle) mode which can achieve a data transfer rate of 33 MB/s
- Cycle translation performed by the system board enables a 32-bit or 16-bit EISA or ISA master to interface with any one of 5 different slaves (EISA 32/16 burst/16 non burst, ISA 16/8 bit)
- Shareable interrupts; Programmable level or edge trigger
- Automatic configuration by means of an on-board product identification ROM. Manufacturers provide a configuration file to be used at system configuration time to assign system resources.

### I/O ACCESSES AND ADDRESSING

EISA supports slot specific I/O access. Since EISA is backwards compatible with ISA addressing, how EISA partitions address space is relatively complex. Next follows a description of how addressing is implemented and how backwards compatibility with ISA limits each EISA slot I/O space to 1 kByte.

EISA supports 16-bit wide I/O addresses providing a total I/O address range of 64k. This is divided into 16 slots, each having 4k allocated to them. This is shown in *Figure 2*.

The top 4 bits of the address LA15:12 define the slot number and the remaining 12 bits LA11:0 provide a 4k address range per slot.

To provide backwards compatibility with ISA, some of this address range must be lost. This is because ISA supports 10-bit wide I/O addresses, resulting in a total I/O space range of 1 kByte. The first 256 bytes (000H-0FFH) of this 1k is allocated to the system board, and the remaining 768 bytes (100H-3FFH) can be used by ISA expansion boards.

This means ISA expansion boards only need to decode addresses 9-0 and therefore will recognize the address range 100H-3FFH (256 to 1k) in every 1k block of the 64k EISA I/O space. That is, all addresses in the top 768 bytes of every 1k block are aliased to the ISA expansion board I/O space (100H-3FFH).

Therefore EISA expansion boards cannot use these addresses and are limited to the bottom 256 locations of every 1k block of I/O space (the ISA system board only uses 256 locations in the first 1k of I/O space). As each slot covers a 4k range in the 64k I/O space, each slot will be able to use 4 blocks of 256 locations (1k). These are z000H-z0FFH, z400H-z4FFH, z800H-z8FFH and zC00H-zCFFH, as shown in the center column of *Figure 2*. EISA devices must only recognize addresses with bits 8 and 9 low (bottom 256 bytes of every 1k block).

ISA supports another slot specific signal AEN which is driven high to all slots by the system whenever a DMA cycle is in progress, to prevent I/O devices from decoding the I/O address on the bus.

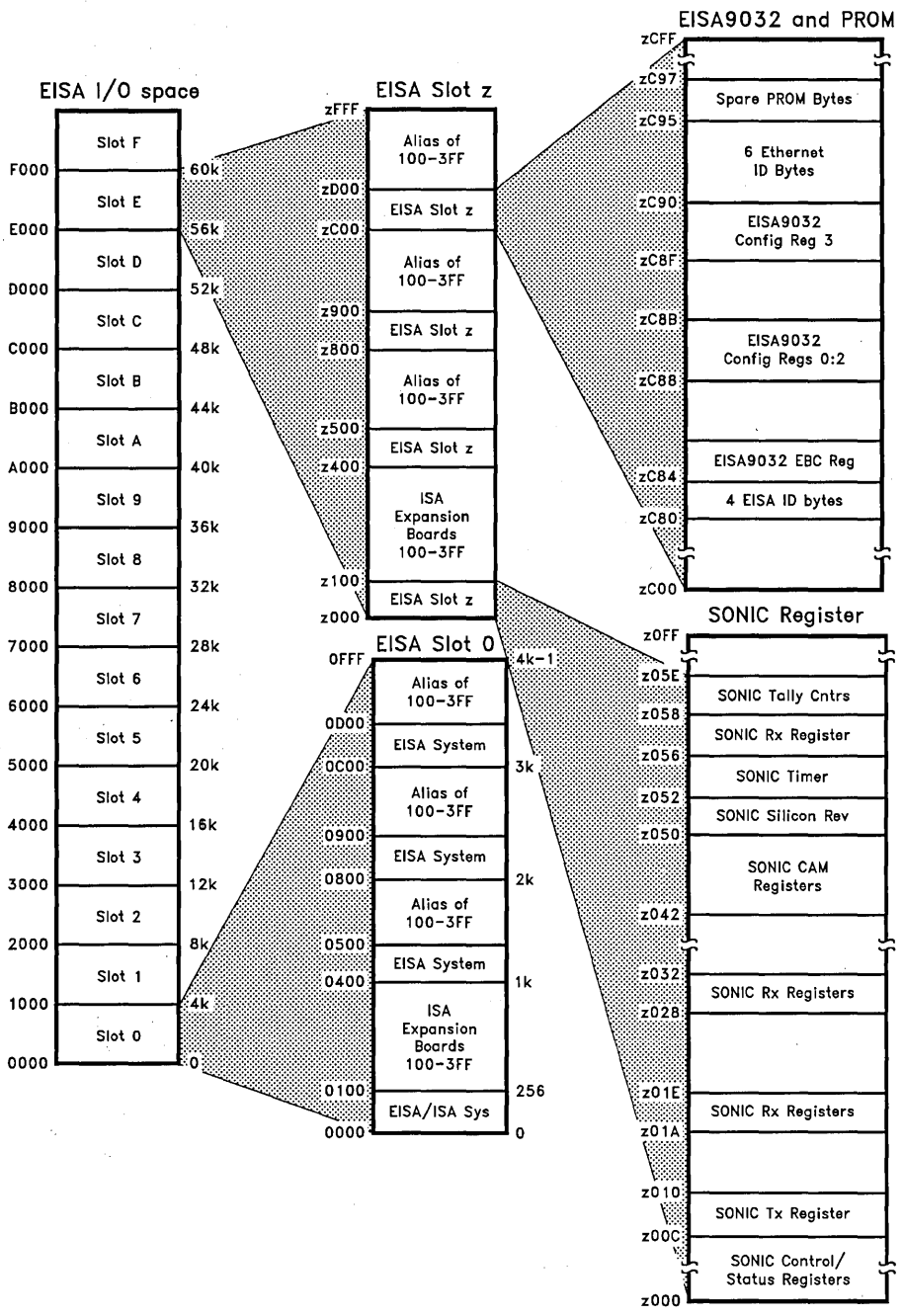


FIGURE 2. EISA I/O Space

In EISA systems, the EISA controller decodes the top four bits of the I/O address LA15:12 (slot number) and only drives AEN active low to the particular slot being accessed. This relieves each slot from having to decode the slot address.

Therefore, EISA devices only need to decode address bits 8 and 9 both low and AEN low to prevent conflict with ISA devices or other EISA slots. This decoding enables the EISA device to use the bottom 256 bytes of its slot address space. The other three 256 byte blocks in its 4k I/O slot space will be aliased to the bottom 256 bytes. To make use of the other three 256 byte blocks and increase its I/O range to 1k, the EISA device must decode address bits 10 and 11.

The SONIC Ethernet adapter card supports 1k of slot specific I/O space decoding (see right column of *Figure 2*). Addresses 0 to 05EH in the first 256 byte block access the SONIC registers.

Addresses 80H to 83H in the last 256 byte block (C00H-CFFH) access the 4 EISA product IDs in the adapter card's PROM. Addresses 90H-97H in the last 256 byte block access the 6 Ethernet ID bytes (plus 2 spare bytes) in the adapter card's PROM. Addresses 84H, 88H to 8BH and 8FH in the last 256 byte block access EISA9032 configuration registers.

PLX's EISA9032 interface chip provides a configuration register bit which enables ISA I/O addressing to be used so that software drivers which used ISA addressing can be used with minor modifications. This board design does not support this configuration, as jumpers would be required to store the I/O base address into the configuration register at power up.

#### EISA BUS ARBITRATION AND BUS LATENCY

EISA provides centralized arbitration control to allow bus sharing between CPU, DMA controller, refresh controller and bus masters. Each master has a slot specific memory request (MRQx) and memory acknowledge (MAKx) signal. If a request is received by the arbitration controller, it will preempt the device currently using the bus who must then release the bus within 64 EISA Bus Clocks (BCKs) (8  $\mu$ s). Therefore a master on the bus can calculate the maximum bus latency (bus request to bus acknowledge delay) it may have to withstand.

EISA supports a three way rotating arbitration priority scheme between refresh, DMA and either the CPU or bus master. The CPU and bus masters maintain a two way rotating arbitration within the original 3 way rotation. For example, if there are two masters and all devices are requesting the bus, this will be the bus acknowledge sequence DMA/ refresh/ CPU/ DMA/ refresh/ Master1/ DMA/ refresh/ CPU/ DMA/ refresh/ Master2. Therefore the worst case bus latency for a bus master with n masters in the system is:

$$(DMA \times 2n) + (\text{refresh} \times 2n) + (CPU \times n) + (\text{master} \times (n-1)) =$$

$$5.8 \mu\text{s} \times 2n + 1.3 \mu\text{s} \times 2n + 9 \times n + 10.6 \times (n-1) = (33.8 \times n - 10.6) \mu\text{s}$$

Therefore for 8 masters = 259.8  $\mu$ s

Note that raising the priority level of a master does not reduce this figure as all other masters must be serviced before the current master can use the bus again. The EISA bus only supports fairness scheme.

The SONIC Ethernet controller will request the bus whenever enough network data has entered its internal FIFO to cross a programmable threshold. The FIFO depth is 32 bytes and the minimum threshold that can be set in the FIFO is 4 bytes. Network data (10 MBits/s) will arrive at 1 byte every 800 ns, therefore the SONIC must acquire the bus before a further 28 bytes arrive into its FIFO, otherwise the FIFO will overflow and the packet will have to be retransmitted. This provides a bus latency of 22.4  $\mu$ s.

#### SYSTEM CONFIGURATION

EISA provides a mechanism for automatic configuration of expansion boards. This eliminates the jumpers required by ISA adapters for board configuration.

The board manufacturer must provide a 4 byte product ID in a PROM which can be read at I/O locations zC80-zC83 and a configuration file with a file name matching the product ID.

At start up the EISA system will read the above I/O locations for every slot and compare the product IDs with what it had stored in non-volatile memory during the last system configuration.

If the system finds a mismatch, the system will need to be reconfigured by running a configuration utility which is provided by each EISA system manufacturer. This utility will look for a configuration file with a name matching the product ID of the board to be installed. The configuration file which is provided by the expansion board manufacturer, contains a list of resources the board is able to use (like interrupt lines for example). The configuration utility will choose which resources to allocate to the board so that it does not conflict with other boards and store the information in non volatile memory.

The board's driver can then read this non volatile memory and program the board so that it will use the resources allocated to it.

The first two bytes of the product ID (locations 0zC80 and 0zC81) contain a compressed representation of the manufacturer's code. The next two bytes (locations 0zC82-3) contain the product number and revision number. Please refer to the EISA specification for details on how these values are derived.

If the expansion board is modified so that it requires a new configuration file, both the product number and revision number must be modified. If it does not require a new configuration file, just the revision number can be changed.

#### EISA BUS PROTOCOL

EISA supports two types of read or write cycles, standard cycles and burst cycles. A burst sequence always starts with a standard cycle. Standard cycles are executed in 2 bus clocks per transfer, whereas burst cycles are executed in 1 bus clock per transfer.

The EISA9032 supports burst read transfers at 25 MHz. At 33 MHz the EISA9032 supports burst read and write transfers. All access to descriptor and resource areas (RRA, RDA and TDA) are executed as standard cycles.

Next follows a description of the standard cycle protocol, how it is converted to a burst cycle sequence, and a brief description of how the EISA9032 interface chip supports these cycles.

For a standard cycle, (see *Figure 3*), once the master has gained control of the bus with the MRQx and MAKx handshake, it initiates a cycle by driving the address and M-I/O signals on the falling edge of the clock (0 to 1 clock transition). On the next rising edge of the clock it drives START for 1 clock period, W/R and BE<3:0> (1 to 2 transition). On the next rising edge of the clock (3 to 4 transition) the system board asserts CMD until the end of the cycle.

The slave, after decoding the address, will drive EX32 active if it can support 32-bit transfers. The master samples this signal on the next rising edge of the clock (3 to 4 transition). If EX32 is not asserted the master will TRI-STATE® its BE<3:0> to enable the system board to perform data size translation. Once the system board has completed the translation it asserts EX32, enabling the master to complete the cycle.

The master then samples the EXRDY line from the slave on the next falling edge of the clock (4 to 5 transition). If it is not asserted the master will insert wait states until EXRDY is asserted. The master can also drive a new address for the next cycle on that same clock edge.

On the next rising edge of the clock (5-6 for a single standard cycle, or 5-2 for back to back standard cycles, or 5-4 for burst cycles) the master or slave will latch the data depending on whether it is a read or write cycle, in this way completing a single standard cycle.

*Figure 4* shows an example of a typical slave access, a SONIC register read.

A burst sequence, (see *Figures 5 and 6*), always starts with a standard cycle which is the protocol described above. If the master wishes to perform a burst of cycles, it will sample the SLBURST signal from the slave during the 3 to 4 clock transition of the initial standard cycle. If the slave has asserted this signal indicating it supports burst cycles, the master will drive MSBURST active which the slave will sample on the last clock edge of the standard cycle (5 to 4 transition).

MSBURST asserted informs the slave that the next cycle is a burst cycle which can be completed in 1 bus clock. The slave will continue to sample MSBURST on every 5-4 clock transition and respond to burst transfers until MSBURST is deasserted. The master or slave will latch the data on the 5-4 clock transition of every transfer depending on whether it is a read or write cycle.

The EISA specification places some restrictions on the use of burst cycles:

1. No I/O cycles
2. No ISA devices
3. No mixed read and write cycles
4. Address lines LA31:10 must remain constant (no crossing of a 1k memory page boundary)

LA9:2 and BE<3:0> can change within a burst, that is addresses don't need to be sequential, and cycle translation and wait states are still supported.

#### Address Pipelining

Note that the EISA protocol requires pipelined addresses, that is the master must provide a new address half a clock

before the data is ready to be latched for the previous cycle if it wants to perform back to back transfers. This is something the SONIC Ethernet controller does not support directly.

For standard cycles this is not a problem as, at the end of a cycle, the EISA9032 interface chip will assert ready to the SONIC, wait for a new address strobe from the SONIC and after driving the new SONIC address on the EISA bus for half a clock, assert the START signal indicating the beginning of a new cycle. This introduces 2 idle bus clock cycles between consecutive standard cycles.

For burst cycles the interface logic must provide a new address during the 4 to 5 clock transition of the previous cycle, as there is no new START signal to indicate when the new address is asserted. This is supported by the EISA9032 interface chip by automatically loading the first address of a burst into an 8-bit counter during the initial standard cycle and incrementing the counter on every 4 to 5 clock transition.

An 8-bit counter for address bits LA9:2 is sufficient, as address lines LA31:10 must remain constant throughout a burst cycle (must not cross a 1k page). The EISA9032 interface chip has a mechanism for detecting when the SONIC address is crossing a 1k page (it detects addresses ending in 3FCH) and will terminate the burst and initiate a new transfer.

Note that because the EISA9032 is using a counter, this means the interface logic only supports bursts to sequential addresses. This is not a problem as burst cycles are only used for the receive and transmit buffer areas which are always addressed sequentially by the SONIC.

Note also that the EISA9032 interface chip starts a cycle in a burst (by driving a new address on the bus and maintaining MSBURST active) before the SONIC has even asserted Address Strobe. This means the interface logic will always do one extra bus cycle at the end of a burst. For read cycles, the software driver must ensure that the end of the TBA is not contiguous to an area of memory that cannot be read. For write cycles, the software driver must ensure that EOBC (End Of Buffer Word Count) in the RBA (Receive Buffer Area) is set at least 2 words larger than the size of the biggest packet that can be received. This means that the SONIC will not use the last two words of an RBA.

#### SLAVE INTERFACE OPERATION

The SONIC Ethernet adapter card supports an EISA slave interface to enable the host CPU to access the following devices on the card.

##### SONIC Registers

The SONIC contains 64 sixteen bit wide registers. Read and write access to 30 of those registers enables the software driver to control and monitor packet transmission and reception. A further 18 registers are used internally by the SONIC. Users may monitor these registers. The last 16 registers (EISA I/O addresses z060H-z07FH) are for test use only. Users must not access these registers. (See *Figure 2* EISA I/O space.)

##### 32-Byte PROM

The adapter card's "EISA product ID" and "Ethernet address" are stored in a 32 x 8 PROM. PROM addresses A0:3 come directly from the EISA bus, but address A4 is generated by the EISA9032 interface chip as the "EISA ID" signal. For EISA I/O addresses 80H-8FH, EISA ID = 1 (EISA

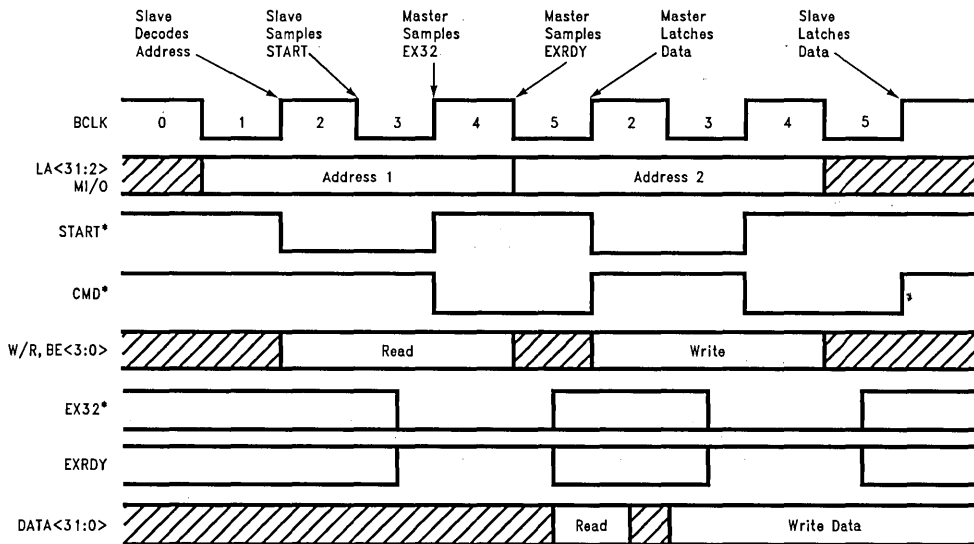


FIGURE 3. EISA Standard Cycle

TL/F/11788-3

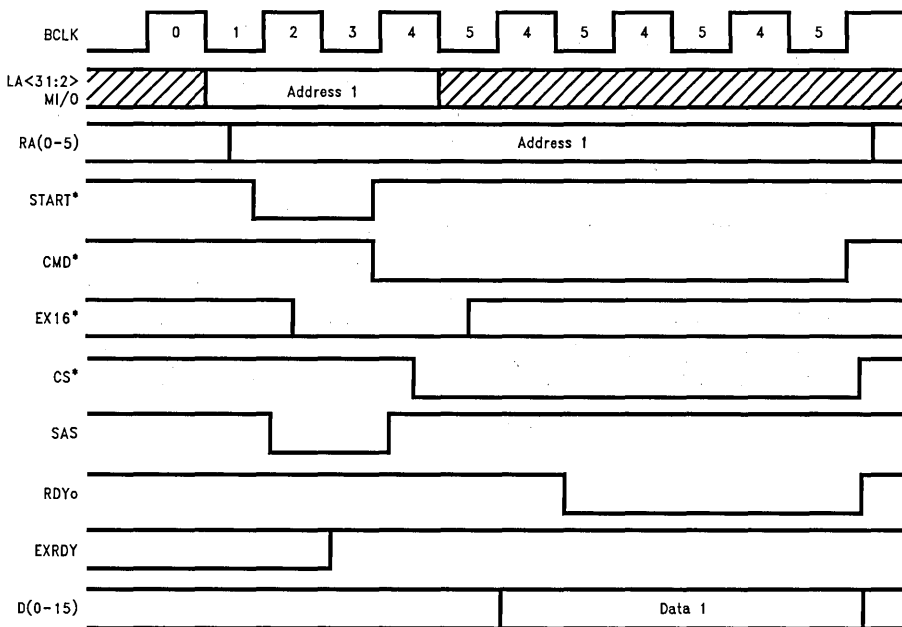


FIGURE 4. SONIC Register Read Cycle

TL/F/11788-4

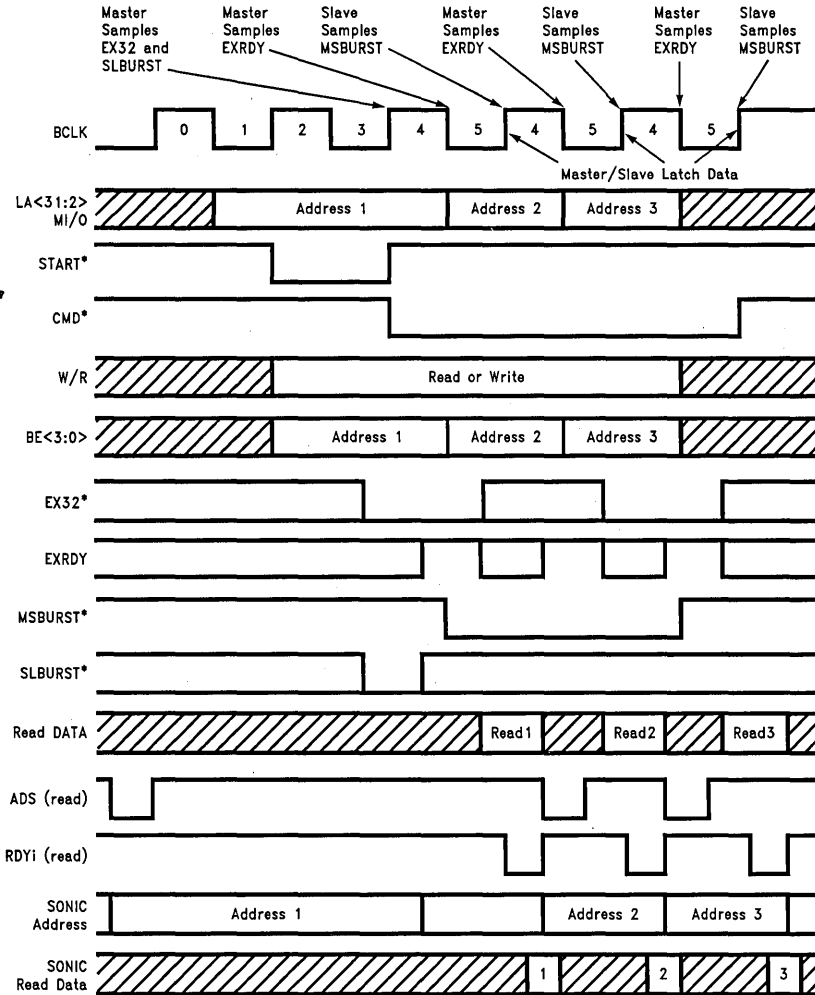


FIGURE 5. EISA Read Burst Cycle

TL/F/11788-5



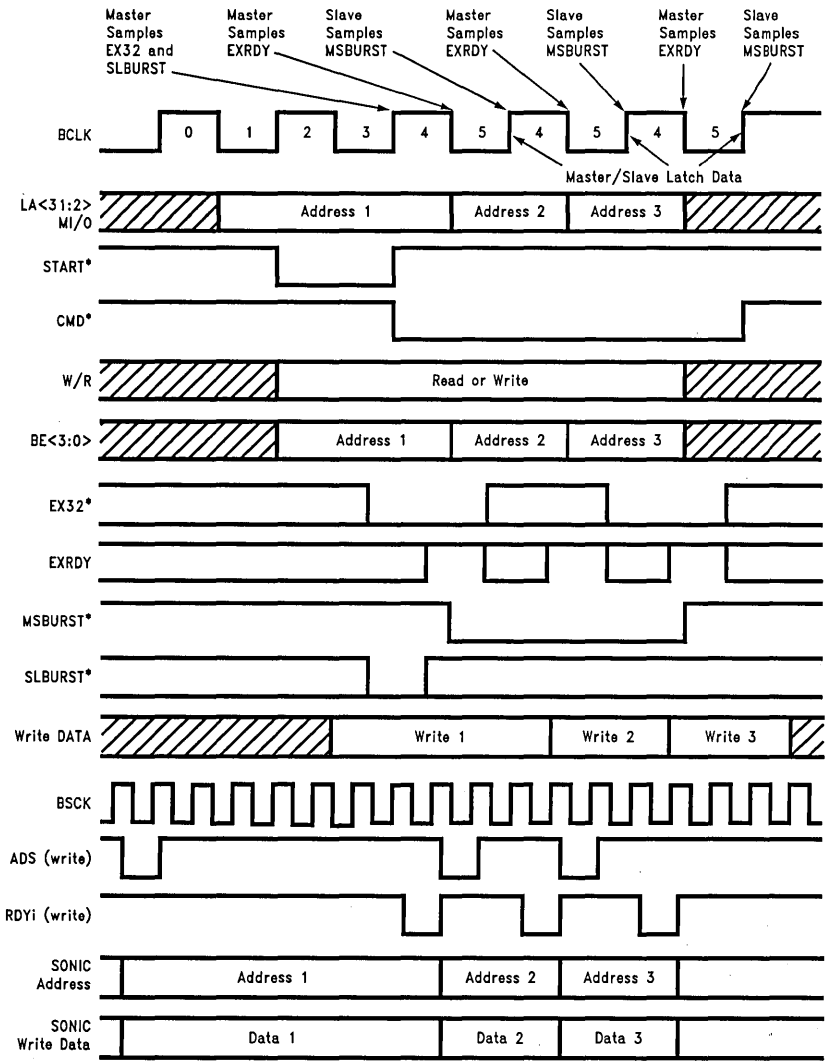


FIGURE 6. EISA Write Burst Cycle

TL/F/11788-6

product ID bytes) and for EISA I/O addresses 90H–97H, EISA ID = 0 (Ethernet ID bytes). This means the EISA9032 chip maps EISA I/O addresses 80H–84H to PROM addresses 10H–14H and EISA addresses 90H–97H to 0H–7H. (Refer to *Figure 2* for I/O map.)

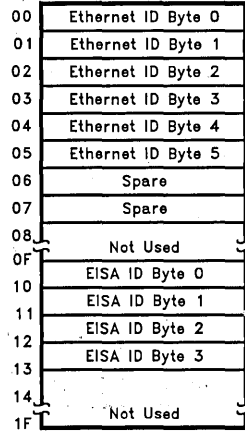
This mapping requires the PROM to be programmed as per *Figure 7*. The first 6 byte locations of this PROM contain the unique physical address assigned to each Ethernet board. These reside on EISA I/O addresses zC90–zC95. The next 2 bytes of the PROM are not used. The following 4 bytes (PROM address 10H–13H) contain the EISA product ID, that is a compressed representation of the manufacturers code, product number and revision number. These 4 bytes reside in EISA I/O space zC80–zC83. The remaining 12 PROM byte locations are not used.

**EISA9032 Bus Interface Configuration Registers**

These registers reside in EISA I/O space zC84H and zC88H–zC8BH.

When configuring the card, the configuration utility program displays a screen enabling the user to select a number of options. The network software driver will then set up the EISA9032 configuration registers according to the values selected by the configuration utility and the user.

Table I lists the configuration options programmable in the EISA9032 registers (also refer to the EISA9032 data sheet).



TL/F/11788-7

**FIGURE 7. Ethernet/EISA ID PROM**

**TABLE I. EISA 9032 Configuration Options**

| Configuration               | Options                  | Selection (Default)          |
|-----------------------------|--------------------------|------------------------------|
| Expansion Board Enable      | Enable/Disable           | (Enable)                     |
| Interrupt Type              | Edge/Level Triggered     | User Selects                 |
| Interrupt Number            | EISA IRQ 5, 9, 10, 11    | EISA Config. Utility Selects |
| Preempt Time                | 55/23 EISA Bus Clocks    | User Selects (23)            |
| Bus Master Data Size        | 32/16 Bits               | (32)                         |
| Slave I/O Data Size         | 32/16 Bits               | (16)                         |
| I/O Addressing              | ISA/Slot Specific        | Slot Specific                |
| ISA I/O Range               |                          | Not Used                     |
| BIOS EPROM Size             | Disable/8k/16k/32k       |                              |
| BIOS EPROM Address Range    |                          | EISA Config. Utility Selects |
| SONIC Register Port Address |                          | Not Used                     |
| Burst Transfer Enable       | Enable/Disable           | (Enable)                     |
| Local Software Reset        | Resets EISA9032          |                              |
| 800 ns Bus Release Timer    | Enable/Disable           | User Selects (Disable)       |
| USR0 ACT/OWN                | Accept/Reject Own Packet | User Selects (Accept)        |
| USR1 Thin/Thick             | Thin/Thick Ethernet      | User Selects (Thin)          |
| USR2                        |                          | Not Used                     |
| USR3                        |                          | Not Used                     |

A number of these options are selected by the EISA configuration Utility program. During configuration the Utility program will read the configuration file generated by the board manufacturer which lists the options the card can support and write its selection into non-volatile memory. The board's software driver will then read this memory and write the selections into the EISA9032 configuration registers. These options include interrupt request lines and BIOS EPROM memory address range.

Another set of options can be selected by the user but should not be changed from their default values on this board. These include Bus Master Data Size = 32 bits, Slave I/O Data Size = 16 bits, I/O Addressing = Slot specific (See I/O Accesses and addressing), Expansion Board Enable = Enable, BIOS EPROM Size = 32k and burst transfer enable = enable. This last option can be used to disable burst transfers so that all the card's master cycles are executed as standard cycles.

A last set of options are system or software dependent and should be selected by the user. These include Interrupt Type (Edge/Level). Level triggered interrupts enable several masters to share an interrupt line. Preempt time of 23 or 55 EISA Bus clocks. This is the number of clocks the SONIC Ethernet card will stay on the EISA bus after the memory acknowledge signal has been deasserted by the arbitrator. Accept/Reject own packet. If in reject mode, the EISA9032 will drive the packet reject input of the SONIC whenever the SONIC is transmitting a packet. Thin/Thick Ethernet will select either Thin or Thick Ethernet by turning the -9V DC-DC converter output to the Coaxial Transceiver Interface on or off.

### 32k x 8 BIOS EPROM

The optional 32k x 8 EPROM design can be added if the user wishes to provide software to boot up the EISA PC from the network. The boot ROM code is simply a special driver that is executed when the EISA PC is initializing, and causes the PC's Operating System to be loaded in from a network server rather than from the EISA PC's hard disk. This software is not provided by National. It can be created by obtaining Novell's Boot ROM developer's kit, or Microsoft's NDDK (Network Device Driver Kit) and following their programming information.

The PROM resides in memory space in the range 0C0000H-0DFFFFH. Its exact location within this range is selected by the EISA configuration utility during board configuration. The card only decodes addresses 17-23.

The Ethernet adapter board supports 6 different types of slave cycle EPROM read, ROM read, SONIC registers read and write, and EISA9032 configuration registers read and write cycles.

### Slave Cycle

EISA slave cycles are initiated by the host CPU driving the 16-bit I/O address or the 24-bit EPROM memory address on the bus and the M-IO signal on the falling edge of the clock, and driving START active with W/R and BE<3:0> on the next rising edge of the clock. The EISA9032 interface chip will decode the address and drive EX16\* low if the CPU is accessing the SONIC registers or its internal registers. It will then drive EXRDY inactive if it needs to insert wait states until the device accessed is ready to provide or accept the data. Bus transfers to the EISA 9032 configuration

registers or the ROM (35 ns access time) are completed with no wait states. Bus transfers to the EPROM (250 ns access time) are completed with two clock cycle wait states and bus transfers to the SONIC will be wait stated until the SONIC asserts RDY<sub>0</sub> indicating it has completed the transfer.

### MASTER INTERFACE DESCRIPTION

The card's main function is to transfer Ethernet packet data from the host's CPU system memory to the Ethernet cable during packet transmission, and from the Ethernet cable to the system memory during packet reception.

The SONIC Ethernet controller's bus master capabilities and buffer management scheme enable it to perform this function using the on board PLX EISA9032 interface chip with no CPU involvement.

Whenever a packet transmission has been requested by the software driver writing to the transmit bit in the command register of the SONIC, or a packet reception is taking place on the Ethernet cable, the SONIC needs to execute read and write cycles on the EISA bus to access descriptor or resource pointer areas in system memory (RDA, TDA, RRA) and to transfer packet data between its internal 32 byte FIFO and buffer areas in system memory (RBA, TBA).

The SONIC initiates a master bus cycle by driving its bus request signal HOLD to the EISA9032 interface chip, who will in turn assert MREQ<sub>x</sub> on the EISA bus. Once the system EISA arbitration controller grants the bus by asserting MAK<sub>x</sub>, the EISA9032 acknowledges the SONIC by driving HLDA so it can start executing a bus cycle. That is the arbitration phase of the bus transfer.

The SONIC then drives the address and status lines to define which area of memory it wishes to access (RRA, RBA, RDA, TDA, TBA) and qualifies them with address strobe ADS. The EISA9032 loads the lower 8 bits of the address A9:2 into its internal counter and initiates a standard cycle on the bus. If during this cycle the interface chip encounters the following conditions, it will drive MSBURST active to initiate a burst cycle following the standard cycle. The conditions are that the SONIC status lines indicate an access to the RBA or TBA, the slave has asserted EX32 indicating it supports 32-bit transfers, the slave has asserted SLBURST indicating it supports burst transfers, the SONIC address does not end in 3FCH (indicating a 1k page crossing) and burst mode was enabled during the adapter card's configuration. The EISA9032 will drive RDY<sub>i</sub> back to the SONIC at the end of every burst cycle.

Note that EISA burst cycles are completed in one EISA clock, whereas SONIC cycles are completed in 3 SONIC bus clocks (asynchronous mode). Therefore for the SONIC to support EISA burst mode it must be run at three times the EISA bus clock speed (25 MHz).

If any of the above conditions were not met, the EISA9032 will assert RDY<sub>i</sub> to the SONIC and wait for a new address strobe before initiating a new standard cycle on the bus.

### DP83932EB-EISA PERFORMANCE

Packet throughput is an important consideration in developing an Ethernet adapter. However, bench-marking of throughput may not tell the whole network performance story. In spite of this, *Figure 8* attempts to compare the performance of this SONIC implementation to other EISA implementations.

In *Figure 8* two tests are shown using NetWare 3.1, and Novell's Perform2 v2.3 performance utility. In both graphs the read/write performance in two configurations using a 4096 byte record size. Both tests use a single 33 MHz 486 server. In *Figure 8a* a single 12 MHz 286 client was used, and in *Figure 8b* 10 8 MHz 286 clients were used.

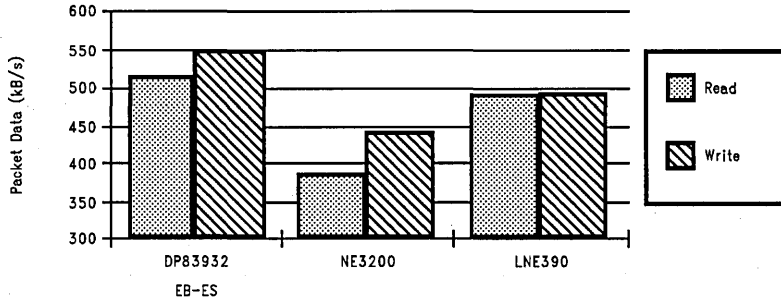
As can be seen from this *Figure*, the performance of the DP83932EB-EISA card surpasses other popular implementations.

**MORE INFORMATION**

For more information regarding the EISA9032, and manufacturing information for the Evaluation board contact:

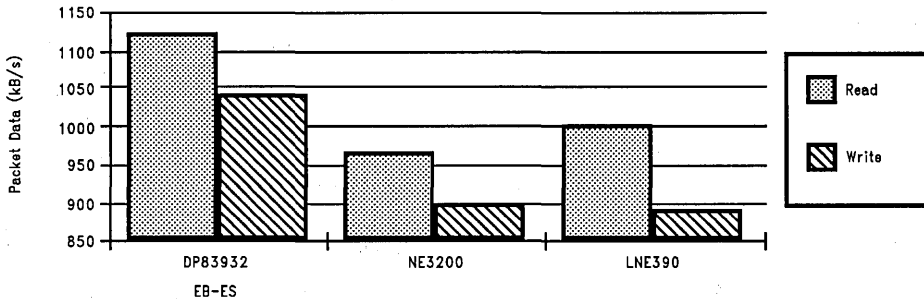
PLX Technology  
 625 Clyde Ave.  
 Mountain View, CA 94043  
 415-960-0448

To obtain the EISA specification contact:  
 BCPR Services 202-371-5921



(a). Single Workstation Performance

TL/F/11788-8



(b). Ten Workstation Performance

TL/F/11788-9

**FIGURE 8. Network Performance Comparison for Single and 10 Workstation LANs**



## Section 2

### ETHERNET PROTOCOL PRODUCTS

#### **10 Mb/s Physical Layer Transceivers and ENDECs**



## Section 2 Contents

|                                                                                |      |
|--------------------------------------------------------------------------------|------|
| DP8392C/DP8392C-1 CTI Coaxial Transceiver Interface .....                      | 2-3  |
| Reliability Data Summary for DP8392 .....                                      | 2-13 |
| DP83910A CMOS SNI Serial Network Interface .....                               | 2-15 |
| DP8391A/NS32491A SNI Serial Network Interface .....                            | 2-25 |
| AN-442 Ethernet/Cheapernet Physical Layer Made Easy with DP8391/92 .....       | 2-35 |
| AN-620 Interfacing the DP8392 to 93 $\Omega$ and 75 $\Omega$ Cable .....       | 2-44 |
| AN-622 Low Power Ethernet with the CMOS DP83910 Serial Network Interface ..... | 2-47 |
| AN-757 Measuring Ethernet Tap Capacitance .....                                | 2-54 |
| Ethernet Magnetics Vendors for 10BASE-T, 10BASE2 and 10BASE5 .....             | 2-57 |

# DP8392C/DP8392C-1 CTI Coaxial Transceiver Interface

## General Description

The DP8392C Coaxial Transceiver Interface (CTI) is a coaxial cable line driver/receiver for Ethernet/Thin Ethernet (Cheapernet) type local area networks. The CTI is connected between the coaxial cable and the Data Terminal Equipment (DTE). In Ethernet applications the transceiver is usually mounted within a dedicated enclosure and is connected to the DTE via a transceiver cable. In Cheapernet applications, the CTI is typically located within the DTE and connects to the DTE through isolation transformers only. The CTI consists of a Receiver, Transmitter, Collision Detector, and a Jabber Timer. The Transmitter connects directly to a 50 ohm coaxial cable where it is used to drive the coax when transmitting. During transmission, a jabber timer is initiated to disable the CTI transmitter in the event of a longer than legal length data packet. Collision Detection circuitry monitors the signals on the coax to determine the presence of colliding packets and signals the DTE in the event of a collision.

The CTI is part of a three chip set that implements the complete IEEE 802.3 compatible network node electronics as shown below. The other two chips are the DP8391 Serial Network Interface (SNI) and the DP8390 Network Interface Controller (NIC).

The SNI provides the Manchester encoding and decoding functions; whereas the NIC handles the Media Access Protocol and the buffer management tasks. Isolation between the CTI and the SNI is an IEEE 802.3 requirement that can be easily satisfied on signal lines using a set of pulse transformers that come in a standard DIP. However, the power isolation for the CTI is done by DC-to-DC conversion through a power transformer.

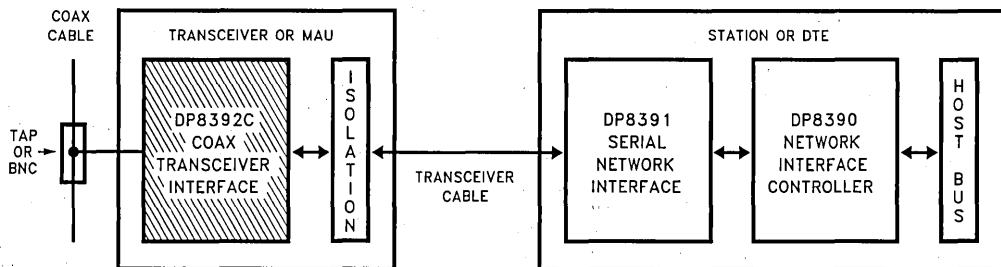
## Features

- Compatible with Ethernet II, IEEE 802.3 10Base5 and 10Base2 (Cheapernet)
- Integrates all transceiver electronics except signal & power isolation
- Innovative design minimizes external component count
- Jabber timer function integrated on chip
- Externally selectable CD Heartbeat allows operation with IEEE 802.3 compatible repeaters
- Precision circuitry implements receive mode collision detection
- Squelch circuitry at all inputs rejects noise
- Designed for rigorous reliability requirements of IEEE 802.3
- Standard Outline 16-pin DIP uses a special leadframe that significantly reduces the operating die temperature

## Table of Contents

- 1.0 System Diagram
- 2.0 Block Diagram
- 3.0 Functional Description
  - 3.1 Receiver Functions
  - 3.2 Transmitter Functions
  - 3.3 Collision Functions
  - 3.4 Jabber Functions
- 4.0 Typical Applications
- 5.0 Connection Diagrams
- 6.0 Pin Descriptions
- 7.0 Absolute Maximum Ratings
- 8.0 DP8392C Electrical Characteristics
- 9.0 DP8392C-1 Electrical Characteristics
- 10.0 Switching Characteristics
- 11.0 Timing and Load Diagram

## 1.0 System Diagram



IEEE 802.3 Compatible Ethernet/Cheapernet Local Area Network Chip Set

TL/F/11085-1

## 2.0 Block Diagram

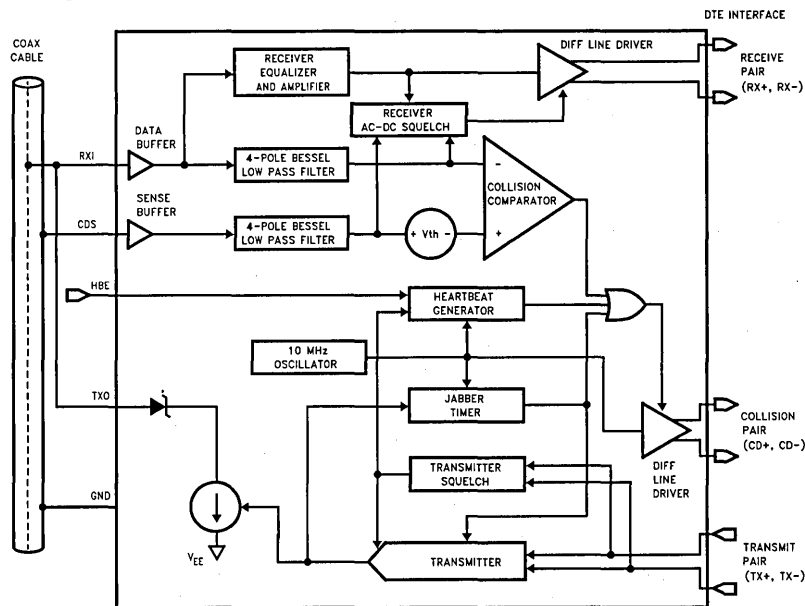


FIGURE 1. DP8392C Block Diagram

TL/F/11085-2

## 3.0 Functional Description

The CTI consists of four main logical blocks:

- the Receiver - receives data from the coax and sends it to the DTE
- the Transmitter - accepts data from the DTE and transmits it onto the coax
- the Collision Detect circuitry - indicates to the DTE any collision on the coax
- the Jabber Timer - disables the Transmitter in case of longer than legal length packets

### 3.1 RECEIVER FUNCTIONS

The Receiver includes an input buffer, a cable equalizer, a 4-pole Bessel low pass filter, a squelch circuit, and a differential line driver.

The buffer provides high input impedance and low input capacitance to minimize loading and reflections on the coax.

The equalizer is a high pass filter which compensates for the low pass effect of the cable. The composite result of the maximum length cable and the equalizer is a flatband response at the signal frequencies to minimize jitter.

The 4-pole Bessel low pass filter extracts the average DC level on the coax, which is used by both the Receiver squelch and the collision detection circuits.

The Receiver squelch circuit prevents noise on the coax from falsely triggering the Receiver in the absence of the signal. At the beginning of the packet, the Receiver turns on when the DC level from the low pass filter is lower than the DC squelch threshold. However, at the end of the packet, a quick Receiver turn off is needed to reject dribble bits. This is accomplished by an AC timing circuit that reacts to high level signals of greater than typically 200 ns in duration. The

Receiver then stays off only if within about 1  $\mu$ s, the DC level from the low pass filter rises above the DC squelch threshold. *Figure 2* illustrates the Receiver timing.

The differential line driver provides ECL compatible signals to the DTE with typically 3 ns rise and fall times. In its idle state, its outputs go to differential zero to prevent DC standing current in the isolation transformer.

### 3.2 TRANSMITTER FUNCTIONS

The Transmitter has a differential input and an open collector output current driver. The differential input common mode voltage is established by the CTI and should not be altered by external circuitry. The transformer coupling of TX $\pm$  will satisfy this condition. The driver meets all IEEE 802.3/Ethernet Specifications for signal levels. Controlled rise and fall times (25 ns V  $\pm$  5 ns) minimize the higher harmonic components. The rise and fall times are matched to minimize jitter. The drive current levels of the DP8392C meet the tighter recommended limits of IEEE 802.3 and are set by a built-in bandgap reference and an external 1% resistor. An on chip isolation diode is provided to reduce the Transmitter's coax load capacitance. For Ethernet compatible applications, an external isolation diode (see *Figure 4*) may be added to further reduce coax load capacitance. In Cheapernet compatible applications the external diode is not required as the coax capacitive loading specifications are relaxed.

The Transmitter squelch circuit rejects signals with pulse widths less than typically 20 ns (negative going), or with levels less than -175 mV. The Transmitter turns off at the end of the packet if the signal stays higher than -175 mV for more than approximately 300 ns. *Figure 3* illustrates the Transmitter timing.



### 3.0 Functional Description (Continued)

#### 3.3 COLLISION FUNCTIONS

The collision circuitry consists of two buffers, two 4-pole Bessel low pass filters (section 3.1), a comparator, a heartbeat generator, a 10 MHz oscillator, and a differential line driver.

Two identical buffers and 4-pole Bessel low pass filters extract the DC level on the center conductor (data) and the shield (sense) of the coax. These levels are monitored by the comparator. If the data level is more negative than the sense level by at least the collision threshold ( $V_{th}$ ), the collision output is enabled.

At the end of every transmission, the heartbeat generator creates a pseudo collision for a short time to ensure that the collision circuitry is properly functioning. This burst on collision output occurs typically  $1.1 \mu s$  after the transmission, and has a duration of about  $1 \mu s$ . This function can be disabled externally with the HBE (Heartbeat Enable) pin to allow operation with repeaters.

The 10 MHz oscillator generates the signal for the collision and heartbeat functions. It is also used as the timebase for all the jabber functions. It does not require any external components.

The collision differential line driver transfers the 10 MHz signal to the  $CD \pm$  pair in the event of collision, jabber, or heartbeat conditions. This line driver also features zero differential idle state.

#### 3.4 JABBER FUNCTIONS

The Jabber Timer monitors the Transmitter and inhibits transmission if the Transmitter is active for longer than 20 ms (fault). It also enables the collision output for the fault duration. After the fault is removed, The Jabber Timer waits for about 500 ms (unjab time) before re-enabling the Transmitter. The transmit input must stay inactive during the unjab time.

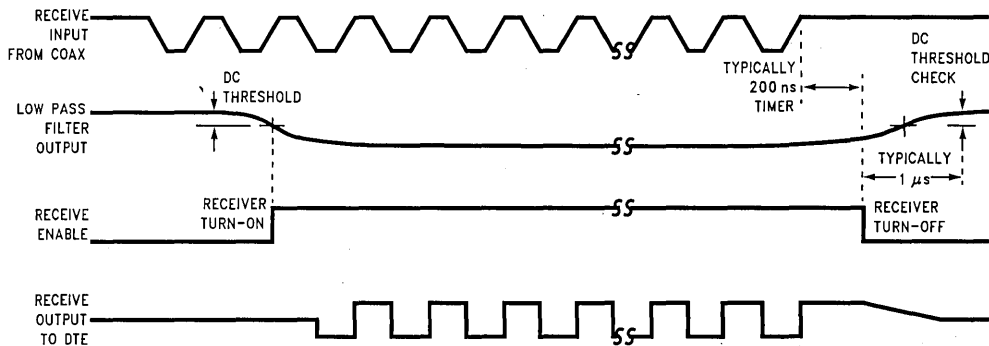


FIGURE 2. Receiver Timing

TL/F/11085-3

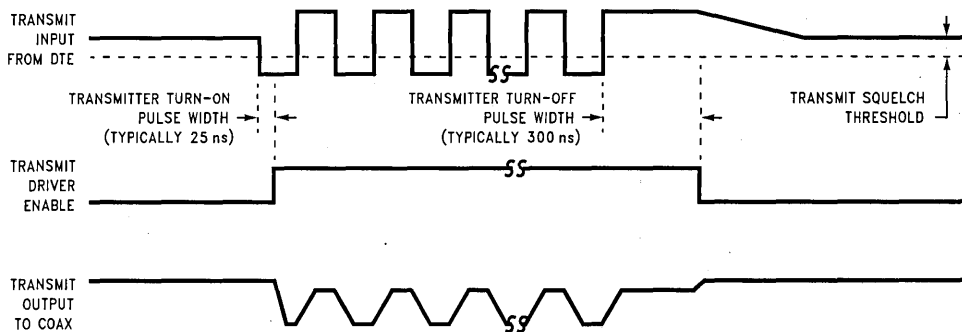
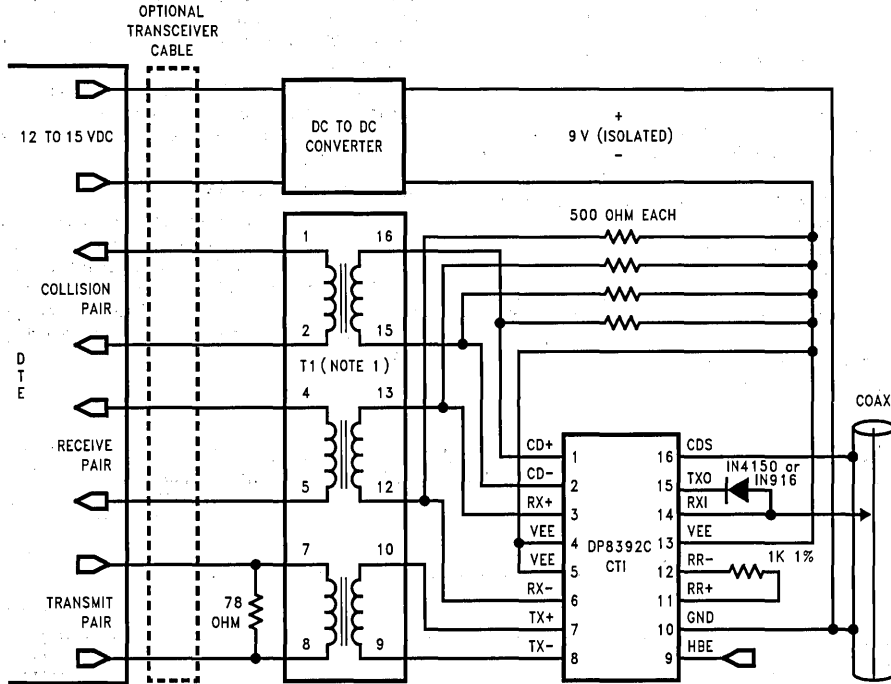


FIGURE 3. Transmitter Timing

TL/F/11085-4

### 4.0 Typical Application

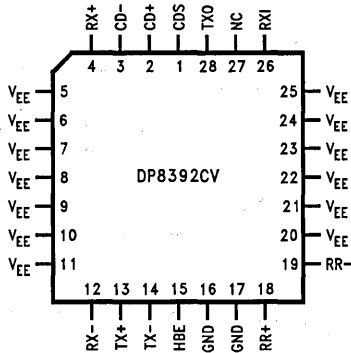


**Note 1:** T1 is a 1:1 pulse transformer, L = 100  $\mu$ H  
 Pulse Engineering (San Diego) Part No. 64103  
 Valor Electronics (San Diego) Part No. LT6003 or equivalent

FIGURE 4

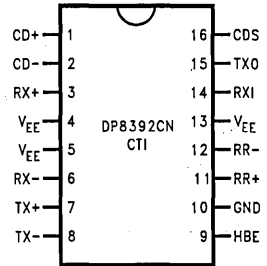
TL/F/11085-5

### 5.0 Connection Diagrams



Order Number DP8392CV  
 See NS Package Number V28A

TL/F/11085-6



Top View  
 Order Number DP8392CN  
 See NS Package Number N16E

TL/F/11085-16

FIGURE 5

## 6.0 Pin Descriptions

| 28-Pin PLCC   | 16-Pin DIP   | Name        | I/O | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------|--------------|-------------|-----|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 2<br>3        | 1<br>2       | CD+*<br>CD- | O   | <b>Collision Output.</b> Balanced differential line driver outputs from the collision detect circuitry. The 10 MHz signal from the internal oscillator is transferred to these outputs in the event of collision, excessive transmission (jabber), or during CD Heartbeat condition. These outputs are open emitters; pulldown resistors to VEE are required. When operating into a 78Ω transmission line, these resistors should be 500Ω. In Cheapernet applications, where the 78Ω drop cable is not used, higher resistor values (up to 1.5k) may be used to save power. |
| 4<br>12       | 3<br>6       | RX+*<br>RX- | O   | <b>Receive Output.</b> Balanced differential line driver outputs from the Receiver. These outputs also require 500Ω pulldown resistors.                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| 13<br>14      | 7<br>8       | TX+*<br>TX- | I   | <b>Transmit Input.</b> Balanced differential line receiver inputs to the Transmitter. The common mode voltage for these inputs is determined internally and must not be externally established. Signals meeting Transmitter squelch requirements are waveshaped and output at TXO.                                                                                                                                                                                                                                                                                          |
| 15            | 9            | HBE         | I   | <b>Heartbeat Enable.</b> This input enables CD Heartbeat when grounded, disables it when connected to VEE.                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 18<br>19      | 11<br>12     | RR+<br>RR-  | I   | <b>External Resistor.</b> A fixed 1k 1% resistor connected between these pins establishes internal operating currents.                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| 26            | 14           | RXI         | I   | <b>Receive Input.</b> Connects directly to the coaxial cable. Signals meeting Receiver squelch requirements are equalized for inter-symbol distortion, amplified, and outputted at RX±.                                                                                                                                                                                                                                                                                                                                                                                     |
| 28            | 15           | TXO         | O   | <b>Transmit Output.</b> Connects either directly (Cheapernet) or via an isolation diode (Ethernet) to the coaxial cable.                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| 1             | 16           | CDS         | I   | <b>Collision Detect Sense.</b> Ground sense connection for the collision detect circuit. This pin should be connected separately to the shield to avoid ground drops from altering the receive mode collision threshold.                                                                                                                                                                                                                                                                                                                                                    |
| 16, 17        | 10           | GND         |     | <b>Positive Supply Pin.</b> A 0.1 μF ceramic decoupling capacitor must be connected across GND and VEE as close to the device as possible.                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| 5-11<br>20-25 | 4<br>5<br>13 | VEE         |     | <b>Negative Supply Pins.</b> In order to make full use of the 3.5W power dissipation capability of this package, these pins should be connected to a large metal frame area on the PC board. Doing this will reduce the operating die temperature of the device thereby increasing the long term reliability.                                                                                                                                                                                                                                                               |

\*IEEE names for CD± = Cl±, RX± = Dl±, TX± = DO±

### 6.1 P.C. BOARD LAYOUT

The DP8392C package is uniquely designed to ensure that the device meets the 1 million hour Mean Time Between Failure (MTBF) requirement of the IEEE 802.3 standard. In order to fully utilize this heat dissipation design, the three VEE pins are to be connected to a copper plane which should be included in the printed circuit board layout.

There are two basic considerations in designing a PCB for the DP8392C and C-1 CTI. The first is ensuring that the layout does not degrade the electrical characteristics of the DP8392, and enables the end product to meet the IEEE 802.3 specifications. The second consideration is meeting the thermal requirements to the DP8392.

Since the DP8392 is highly integrated the layout is actually quite simple, and there are just a few guidelines:

1. Ensure that the parasitic capacitance added to the RXI and TXO pins is minimized. To do this keep these signal traces short, and remove any power planes under these signals, and under any components that connect to these signals. *Figure 6* shows the component placement for the DIP package. The PLCC component placement would be similar, as shown in *Figure 7*.

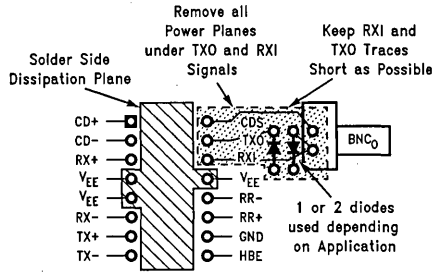
2. The power supply layout to the CTI should be relatively clean. Usually the CTI's power is supplied directly by a DC-DC converter. The power should be routed either through separate isolated planes, or via thick PCB traces.

For the second consideration, the packaged DP8392 must have a thermal resistance of 40°C–45°C/W to meet the full 0°C–70°C temperature range. The CTI dissipates more power when transmitting than while it is idle. In order to do this the thermal resistance of the device must be 40°C–45°C/W. To meet this requirement during transmission, it is recommended that a small printed circuit board plane be connected to all VEE pins on the solder side of the PCB.

The size of the trace plane depends on the package used and the duty cycle of transmissions. For the DIP package the plane should be connected to pins 4–5, 13, and the size should be approximately 0.2 square inches for applications where the duty cycle of the transmitter is very low (<10%). This would be typical of adapter or motherboard applications. In applications where the transmitter duty cycle may be large (repeaters and external transceivers) the total area should be increased to 0.4 in<sup>2</sup>. *Figure 6* illustrates a recommended component side layout for these planes.

### 6.0 Pin Descriptions (Continued)

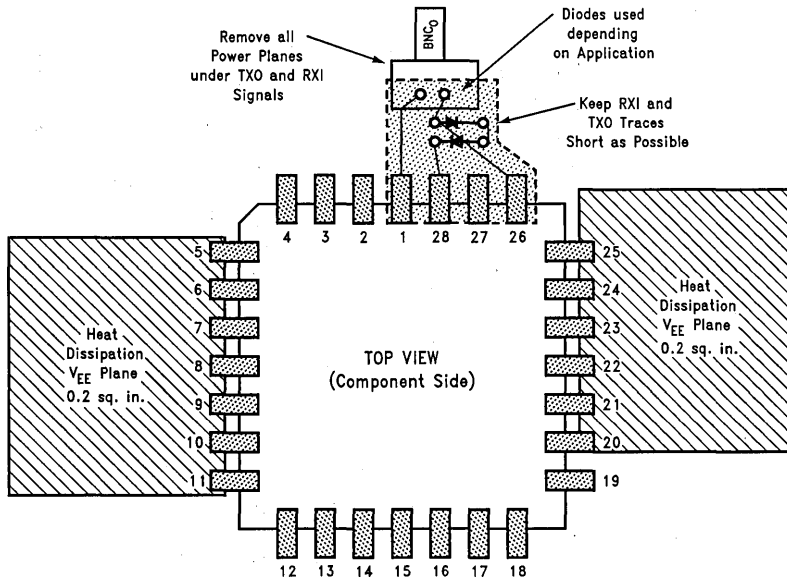
For the PLCC packaged DP8392, it is recommended that a small printed circuit board  $V_{EE}$  plane be connected to pins 5-11, and a second one be connected to pins 20-25. To reduce the thermal resistance to the required value, the area of the plane on EACH set of pins should be  $\geq 0.20 \text{ in}^2$  for applications with low transmitter duty cycle, and  $\geq 0.4 \text{ in}^2$  for high transmit duty cycle applications. Figure 7 illustrates a recommended component side layout for these planes.



TL/F/11085-14

Layout as viewed from component side

**FIGURE 6. Typical Layout Considerations for DP8392CN (Not to Scale)**



TL/F/11085-15

**FIGURE 7. Recommended Layout and Dissipation Planes for DP8392CV (Not to Scale)**

**7.0 Absolute Maximum Ratings** (Note 1)

|                                                    |                             |
|----------------------------------------------------|-----------------------------|
| Supply Voltage ( $V_{EE}$ )                        | -12V                        |
| Package Power Rating at 25°C<br>(PC Board Mounted) | 3.5 Watts*<br>See Section 5 |
| Derate linearly at the rate of 28.6 mW/°C          |                             |
| Input Voltage                                      | 0 to -12V                   |
| Storage Temperature                                | -65° to 150°C               |
| Lead Temp. (Soldering, 10 seconds)                 | 260°C                       |

\*For actual power dissipation of the device please refer to section 7.0.

**Recommended Operating Conditions**

|                             |            |
|-----------------------------|------------|
| Supply Voltage ( $V_{EE}$ ) | -9V ± 5%   |
| Ambient Temperature         | 0° to 70°C |

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

**8.0 DP8392C Electrical Characteristics**  $V_{EE} = -9V \pm 5\%$ ,  $T_A = 0^\circ$  to 70°C (Notes 2 & 3)

All parameters with respect to CD± and RX± are measured after the pulse transformer except  $V_{OC}$ .

| Symbol    | Parameter                                           | Min   | Typ   | Max       | Units |
|-----------|-----------------------------------------------------|-------|-------|-----------|-------|
| $I_{EE1}$ | Supply current out of $V_{EE}$ pin—non transmitting |       | -85   | -130      | mA    |
| $I_{EE2}$ | Supply current out of $V_{EE}$ pin—transmitting     |       | -125  | -180      | mA    |
| $I_{RXI}$ | Receive input bias current (RXI)                    | -2    |       | +25       | μA    |
| $I_{TDC}$ | Transmit output dc current level (TXO)              | 37    | 41    | 45        | mA    |
| $I_{TAC}$ | Transmit output ac current level (TXO)              | ±28   |       | $I_{TDC}$ | mA    |
| $V_{CD}$  | Collision threshold (Receive mode)                  | -1.45 | -1.53 | -1.58     | V     |
| $V_{OD}$  | Differential output voltage (RX±, CD±)              | ±550  |       | ±1200     | mV    |
| $V_{OC}$  | Common mode output voltage (RX±, CD±)               | -1.5  | -2.0  | -2.5      | V     |
| $V_{OB}$  | Diff. output voltage imbalance (RX±, CD±)           |       |       | ±40       | mV    |
| $V_{TS}$  | Transmitter squelch threshold (TX±)                 | -175  | -225  | -300      | mV    |
| $C_X$     | Input capacitance (RXI)                             |       | 1.2   |           | pF    |
| $R_{RXI}$ | Shunt resistance—non transmitting (RXI)             | 100   |       |           | KΩ    |
| $R_{TXO}$ | Shunt resistance—transmitting (TXO)                 |       | 10    |           | KΩ    |

**9.0 DP8392C-1 Electrical Characteristics**  $V_{EE} = -9V \pm 5\%$ ,  $T_A = 0^\circ$  to 70°C (Notes 2 & 3)

All parameters with respect to CD± and RX± are measured after the pulse transformer except  $V_{OC}$ .

| Symbol    | Parameter                                           | Min   | Typ   | Max       | Units |
|-----------|-----------------------------------------------------|-------|-------|-----------|-------|
| $I_{EE1}$ | Supply current out of $V_{EE}$ pin—non transmitting |       | -85   | -130      | mA    |
| $I_{EE2}$ | Supply current out of $V_{EE}$ pin—transmitting     |       | -125  | -180      | mA    |
| $I_{RXI}$ | Receive input bias current (RXI)                    | -2    |       | +25       | μA    |
| $I_{TDC}$ | Transmit output dc current level (TXO)              | 37    | 41    | 45        | mA    |
| $I_{TAC}$ | Transmit output ac current level (TXO)              | ±28   |       | $I_{TDC}$ | mA    |
| $V_{CD}$  | Collision threshold (Receive mode)                  | -1.45 | -1.53 | -1.58     | V     |
| $V_{OD}$  | Differential output voltage (RX±, CD±)              | ±550  |       | ±1200     | mV    |
| $V_{OC}$  | Common mode output voltage (RX±, CD±)               | -1.5  | -2.0  | -2.5      | V     |
| $V_{OB}$  | Diff. output voltage imbalance (RX±, CD±)           |       |       | ±40       | mV    |
| $V_{TS}$  | Transmitter squelch threshold (TX±)                 | -175  | -225  | -275      | mV    |
| $C_X$     | Input capacitance (RXI)                             |       | 1.2   |           | pF    |
| $R_{RXI}$ | Shunt resistance—non transmitting (RXI)             | 100   |       |           | KΩ    |
| $R_{TXO}$ | Shunt resistance—transmitting (TXO)                 | 7.5K  | 10    |           | KΩ    |

**Note 1:** Absolute maximum ratings are those values beyond which the safety of the device cannot be guaranteed. They are not meant to imply that the device should be operated at these limits.

**Note 2:** All currents into device pins are positive, all currents out of device pins are negative. All voltages referenced to ground unless otherwise specified.

**Note 3:** All typicals are given for  $V_{EE} = -9V$  and  $T_A = 25^\circ C$ .

## 10.0 DP8392C Switching Characteristics $V_{EE} = -9V \pm 5\%$ , $T_A = 0^\circ$ to $70^\circ\text{C}$ (Note 3)

| Symbol     | Parameter                                               | Fig     | Min | Typ       | Max  | Units         |
|------------|---------------------------------------------------------|---------|-----|-----------|------|---------------|
| $t_{RON}$  | Receiver startup delay (RXI to RX $\pm$ )               | 8 & 14  |     | 4         |      | bits          |
| $t_{Rd}$   | Receiver propagation delay (RXI to RX $\pm$ )           | 8 & 14  |     | 15        | 50   | ns            |
| $t_{Rr}$   | Differential outputs rise time (RX $\pm$ , CD $\pm$ )   | 8 & 14  |     | 4         |      | ns            |
| $t_{Rf}$   | Differential outputs fall time (RX $\pm$ , CD $\pm$ )   | 8 & 14  |     | 4         |      | ns            |
| $t_{RJ}$   | Receiver & cable total jitter                           | 13      |     | $\pm 2$   |      | ns            |
| $t_{TST}$  | Transmitter startup delay (TX $\pm$ to TXO)             | 9 & 14  |     | 1         |      | bits          |
| $t_{Td}$   | Transmitter propagation delay (TX $\pm$ to TXO)         | 9 & 14  |     | 25        | 50   | ns            |
| $t_{Tr}$   | Transmitter rise time —10% to 90% (TXO)                 | 9 & 14  |     | 25        |      | ns            |
| $t_{Tf}$   | Transmitter fall time —90% to 10% (TXO)                 | 9 & 14  |     | 25        |      | ns            |
| $t_{TM}$   | $t_{Tr}$ and $t_{Tf}$ mismatch                          |         |     | 0.5       |      | ns            |
| $t_{TS}$   | Transmitter skew (TXO)                                  |         |     | $\pm 0.5$ |      | ns            |
| $t_{TON}$  | Transmit turn-on pulse width at $V_{TS}$ (TX $\pm$ )    | 9 & 14  |     | 20        |      | ns            |
| $t_{TOFF}$ | Transmit turn-off pulse width at $V_{TS}$ (TX $\pm$ )   | 9 & 14  |     | 250       |      | ns            |
| $t_{CON}$  | Collision turn-on delay                                 | 10 & 14 |     | 7         |      | bits          |
| $t_{COFF}$ | Collision turn-off delay                                | 10 & 14 |     |           | 20   | bits          |
| $f_{CD}$   | Collision frequency (CD $\pm$ )                         | 10 & 14 | 8.0 |           | 12.5 | MHz           |
| $t_{CP}$   | Collision pulse width (CD $\pm$ )                       | 10 & 14 | 35  |           | 70   | ns            |
| $t_{HON}$  | CD Heartbeat delay (TX $\pm$ to CD $\pm$ )              | 11 & 14 | 0.6 |           | 1.6  | $\mu\text{s}$ |
| $t_{HW}$   | CD Heartbeat duration (CD $\pm$ )                       | 11 & 14 | 0.5 | 1.0       | 1.5  | $\mu\text{s}$ |
| $t_{JA}$   | Jabber activation delay (TX $\pm$ to TXO and CD $\pm$ ) | 12 & 14 | 20  | 29        | 60   | ms            |
| $t_{JR}$   | Jabber reset unjab time (TX $\pm$ to TXO and CD $\pm$ ) | 12 & 14 | 250 | 500       | 750  | ms            |

## DP8392C-1 Switching Characteristics $V_{EE} = -9V \pm 5\%$ , $T_A = 0^\circ$ to $70^\circ\text{C}$ (Note 3)

| Symbol     | Parameter                                               | Fig     | Min | Typ       | Max  | Units         |
|------------|---------------------------------------------------------|---------|-----|-----------|------|---------------|
| $t_{RON}$  | Receiver startup delay (RXI to RX $\pm$ )               | 8 & 14  |     | 4         | 5    | bits          |
| $t_{Rd}$   | Receiver propagation delay (RXI to RX $\pm$ )           | 8 & 14  |     | 15        | 50   | ns            |
| $t_{Rr}$   | Differential outputs rise time (RX $\pm$ , CD $\pm$ )   | 8 & 14  |     | 4         | 7    | ns            |
| $t_{Rf}$   | Differential outputs fall time (RX $\pm$ , CD $\pm$ )   | 8 & 14  |     | 4         | 7    | ns            |
| $t_{RJ}$   | Receiver & cable total jitter                           | 13      |     | $\pm 2$   |      | ns            |
| $t_{TST}$  | Transmitter startup delay (TX $\pm$ to TXO)             | 9 & 14  |     | 1         | 2    | bits          |
| $t_{Td}$   | Transmitter propagation delay (TX $\pm$ to TXO)         | 9 & 14  | 5   | 25        | 50   | ns            |
| $t_{Tr}$   | Transmitter rise time —10% to 90% (TXO)                 | 9 & 14  | 20  | 25        | 30   | ns            |
| $t_{Tf}$   | Transmitter fall time —90% to 10% (TXO)                 | 9 & 14  | 20  | 25        | 30   | ns            |
| $t_{TM}$   | $t_{Tr}$ and $t_{Tf}$ mismatch                          |         |     | 0.5       |      | ns            |
| $t_{TS}$   | Transmitter skew (TXO)                                  |         |     | $\pm 0.5$ |      | ns            |
| $t_{TON}$  | Transmit turn-on pulse width at $V_{TS}$ (TX $\pm$ )    | 9 & 14  | 5   | 20        | 40   | ns            |
| $t_{TOFF}$ | Transmit turn-off pulse width at $V_{TS}$ (TX $\pm$ )   | 9 & 14  | 110 |           | 270  | ns            |
| $t_{CON}$  | Collision turn-on delay                                 | 10 & 14 |     | 7         | 13   | bits          |
| $t_{COFF}$ | Collision turn-off delay                                | 10 & 14 |     |           | 20   | bits          |
| $f_{CD}$   | Collision frequency (CD $\pm$ )                         | 10 & 14 | 8.5 |           | 12.5 | MHz           |
| $t_{CP}$   | Collision pulse width (CD $\pm$ )                       | 10 & 14 | 35  |           | 70   | ns            |
| $t_{HON}$  | CD Heartbeat delay (TX $\pm$ to CD $\pm$ )              | 11 & 14 | 0.6 |           | 1.6  | $\mu\text{s}$ |
| $t_{HW}$   | CD Heartbeat duration (CD $\pm$ )                       | 11 & 14 | 0.5 | 1.0       | 1.5  | $\mu\text{s}$ |
| $t_{JA}$   | Jabber activation delay (TX $\pm$ to TXO and CD $\pm$ ) | 12 & 14 | 20  | 29        | 60   | ms            |
| $t_{JR}$   | Jabber reset unjab time (TX $\pm$ to TXO and CD $\pm$ ) | 12 & 14 | 250 | 500       | 750  | ms            |

**Note 1:** Absolute maximum ratings are those values beyond which the safety of the device cannot be guaranteed. They are not meant to imply that the device should be operated at these limits.

**Note 2:** All currents into device pins are positive, all currents out of device pins are negative. All voltages referenced to ground unless otherwise specified.

**Note 3:** All typicals are given for  $V_{EE} = -9V$  and  $T_A = 25^\circ\text{C}$ .

# 11.0 Timing and Load Diagrams

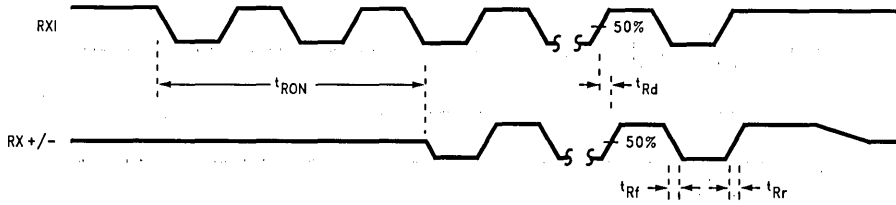


FIGURE 8. Receiver Timing

TL/F/11085-7

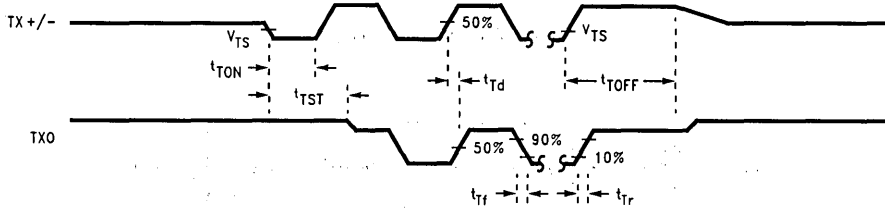


FIGURE 9. Transmitter Timing

TL/F/11085-8

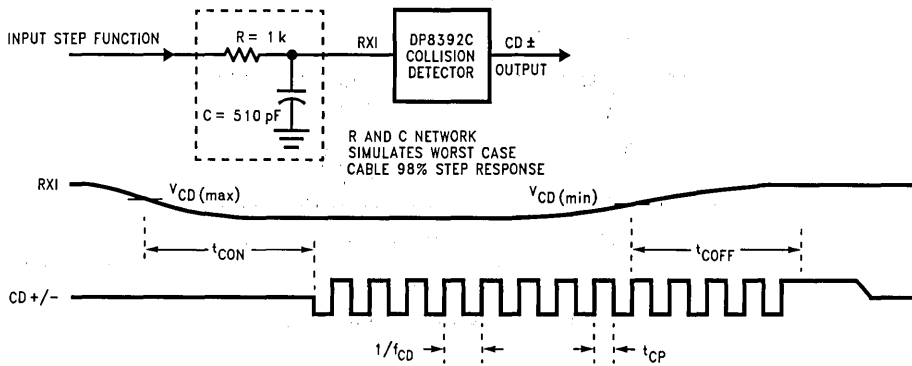


FIGURE 10. Collision Timing

TL/F/11085-9

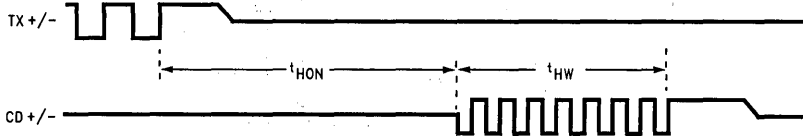


FIGURE 11. Heartbeat Timing

TL/F/11085-10

### 11.0 Timing and Load Diagrams (Continued)

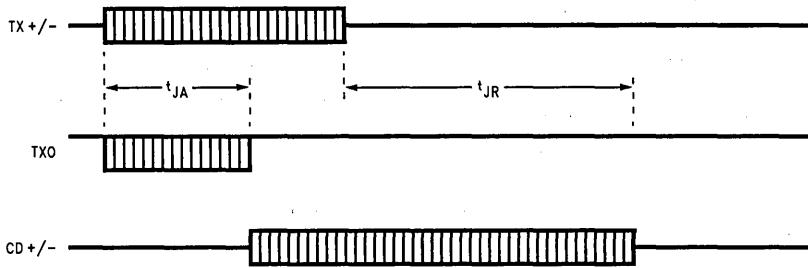
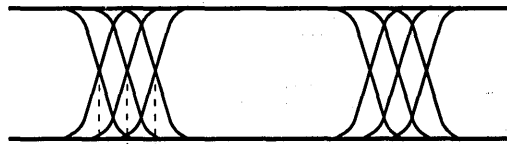
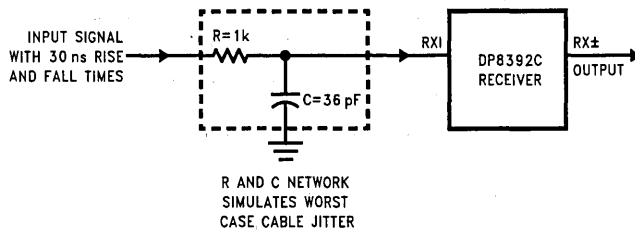


FIGURE 12. Jabber Timing

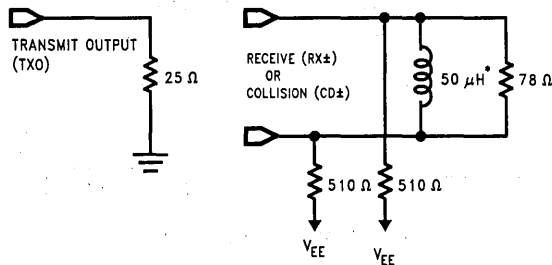
TL/F/11085-11



Receiver equalization (jitter correction)  $\geq 1$  ns  
 Input jitter at RX $\pm$   $\leq \pm 7$  ns  
 Output jitter at RX $\pm$   $\leq \pm 6$  ns

TL/F/11085-12

FIGURE 13. Receive Jitter Timing



TL/F/11085-13

\*The 50  $\mu$ H inductance is for testing purposes. Pulse transformers with higher inductances are recommended (see Figure 4)

FIGURE 14. Test Loads



# Reliability Data Summary for DP8392



## REF: TEST LAB FILES

RDT25406                      RDT26627  
 RDT25500                      RDT26638  
 RDT26562

## TEST SAMPLE DESCRIPTION/HISTORY

| Lot | Device      | Package     | Date Code | Fab Location | Assembly Location |
|-----|-------------|-------------|-----------|--------------|-------------------|
| 1   | DP8392      | N, 16 Leads | 8509      | NSSC         | NSEB              |
| 2   | DP8392      | N, 16 Leads | 8513      | NSSC         | NSEB              |
| 3   | DP8392      | N, 16 Leads | 8526      | NSSC         | NSEB              |
| 4   | DP8392      | N, 16 Leads | 8552      | NSSC         | NSEB              |
| 5   | DP8392A(-4) | N, 16 Leads | 8620      | NSUK         | NSEB              |
| 6   | DP8392A(-5) | N, 16 Leads | 8637      | NSUK         | NSEB              |
| 7   | DP8392A(-5) | N, 16 Leads | 8637      | NSUK         | NSEB              |
| 8   | DP8392A(-5) | N, 16 Leads | 8637      | NSUK         | NSEB              |
| 9   | DP8392C     | N, 16 Leads | 9106      | NSUK         | NSEB              |
| 10  | DP8392C     | N, 16 Leads | 9106      | NSUK         | NSEB              |
| 11  | DP8392C     | N, 16 Leads | 9106      | NSUK         | NSEB              |
| 12  | DP8392C     | N, 16 Leads | 9106      | NSUK         | NSEB              |
| 13  | DP8392C     | N, 16 Leads | 9106      | NSUK         | NSEB              |
| 14  | DP8392C     | N, 16 Leads | 9106      | NSUK         | NSEB              |

## ABSTRACT

DP8392 Coaxial Transceiver Interface parts from 8 lots were subjected to Operating Life Test, Temperature and Humidity Bias Test, Temperature Cycle Test, and Electrostatic Discharge Test.

## PURPOSE OF TEST

Evaluation of new device and qualification of U.K. fab.

## TESTS PERFORMED

- Operating Life Test (OPL) (100°C; biased)
- Operating Life Test (OPL) (125°C; biased)
- Temperature and Humidity Bias Test (THBT) (85°C; 85% R.H.; biased)
- Temperature Cycle Test (TMCL) (-40°C, +125°C; unbiased)
- Electrostatic Discharge Test (ESD) (Human body model: R = 1500Ω; C = 120 pF)

## CONCLUSIONS

1. The DP8392AN exceeds the IEEE 802.3 specification of 1 million hours Mean Time Between Failure (MTBF).
2. U.K. fab results are comparable to those of Santa Clara. On ESD testing all pins passed at 1000V except for pin 7 (TX<sup>+</sup>).

## RESULTS

| Test | Temperature    | Lot | Fab  | Time Point—Number of Failures |      |       |       |       |
|------|----------------|-----|------|-------------------------------|------|-------|-------|-------|
|      |                |     |      | Hours                         |      |       |       |       |
|      |                |     |      | 168                           | 336  | 500   | 1000  | 2000  |
| OPL  | 100°C          | 1   | NSSC | 0/50                          |      | 0/50  | 0/50  |       |
|      | 100°C          | 2   | NSSC | 0/50                          |      | 0/50  | 0/50  |       |
|      | 125°C          | 3   | NSSC | 0/74                          |      |       |       |       |
|      | 125°C          | 4   | NSSC | 0/100                         |      | 0/100 | 0/100 | 0/100 |
|      | 100°C          | 5   | NSUK | 0/60                          |      |       |       |       |
|      | 100°C          | 6   | NSUK |                               | 0/33 | 0/33  | 0/33  | 0/33  |
|      | 100°C          | 7   | NSUK |                               | 0/31 | 0/31  | 0/31  | 0/31  |
|      | 100°C          | 8   | NSUK |                               | 0/33 | 0/31  | 0/31  | 0/31  |
|      | 85°C           | 9   | NSUK |                               |      | 0/77  | 0/77  |       |
|      | 85°C           | 10  | NSUK |                               |      | 0/77  | 0/77  |       |
|      | 85°C           | 11  | NSUK |                               |      | 0/77  | 0/77  |       |
|      | 100°C          | 12  | NSUK | 0/64                          |      | 0/64  | 0/64  | 0/64  |
|      | 100°C          | 13  | NSUK | 0/25                          |      | 0/25  | 0/25  | 0/25  |
|      | 100°C          | 14  | NSUK | 0/10                          |      | 0/10  | 0/10  | 0/10  |
| THBT | 85°C; 85% R.H. | 1   | NSSC | 0/50                          |      | 0/50  | 0/50  |       |
|      |                | 2   | NSSC | 0/50                          |      | 0/50  | 0/50  |       |
|      |                | 3   | NSSC | 0/75                          |      | 0/75  | 0/75  |       |
|      |                | 9   | NSUK | 0/30                          |      | 0/30  | 0/30  |       |
|      |                | 10  | NSUK | 0/30                          |      | 0/30  | 0/30  |       |
|      |                | 11  | NSUK | 0/30                          |      | 0/30  | 0/30  |       |

**RESULTS (Continued)**

| Test | Temperature      | Lot | Fab  | Time Point—Number of Failures |      |      |      |      |
|------|------------------|-----|------|-------------------------------|------|------|------|------|
|      |                  |     |      | Hours                         |      |      |      |      |
|      |                  |     |      | 168                           | 336  | 500  | 1000 | 2000 |
| ACLV | 121°C; 100% R.H. | 9   | NSUK | 0/77                          |      | 0/77 |      |      |
|      |                  | 10  | NSUK | 0/66                          |      | 0/66 |      |      |
|      |                  | 11  | NSUK | 0/77                          |      | 0/77 |      |      |
|      |                  |     |      | Cycles                        |      |      |      |      |
|      |                  |     |      | 500                           | 1000 | 2000 | 3000 |      |
| TMCL | -40°C, +125°C    | 4   | NSSC | 0/70                          | 0/70 | 0/70 | 0/70 |      |
|      | -65°C, +150°C    | 9   | NSUK | 0/77                          | 0/77 | 0/77 |      |      |
|      | -65°C, +150°C    | 10  | NSUK | 0/66                          | 0/66 | 0/66 |      |      |
|      | -65°C, +150°C    | 11  | NSUK | 0/77                          | 0/77 | 0/77 |      |      |

**ELECTROSTATIC DISCHARGE TEST (ESD) RESULTS**

26 parts from 4 wafer lots were tested by the Human Body Model test condition; R = 1500Ω; C = 120 pF. First ground was held common, then V<sub>EE</sub>. 5 positive and 5 negative pulses were applied for each pin/voltage combination.

| Pin | Function        | Voltage—Number of Failures |       |
|-----|-----------------|----------------------------|-------|
|     |                 | 500V                       | 1000V |
| 1   | CD+             | 0/26                       | 0/20  |
| 2   | CD-             | 0/26                       | 0/20  |
| 3   | RX+             | 0/26                       | 0/20  |
| 4   | V <sub>EE</sub> | 0/26                       | 0/20  |
| 5   | V <sub>EE</sub> | 0/26                       | 0/20  |
| 6   | RX-             | 0/26                       | 0/20  |
| 7   | TX+             | 6/26                       | 13/20 |
| 8   | TX-             | 0/26                       | 0/20  |
| 9   | HBE             | 0/26                       | 0/20  |
| 10  | GND             | 0/26                       | 0/20  |
| 11  | RR+             | 0/26                       | 0/20  |
| 12  | V <sub>EE</sub> | 0/26                       | 0/20  |
| 13  | V <sub>EE</sub> | 0/26                       | 0/20  |
| 14  | RXI             | 0/26                       | 0/20  |
| 15  | TXO             | 0/26                       | 0/20  |
| 16  | CDS             | 0/26                       | 0/20  |

Further characterization has been done to determine individual pin ESD damage thresholds. In particular, for pin 7 (TX+), 80 parts from 4 wafer lots were tested. Pin 7 ESD damage thresholds varied from 200V-300V to 2000V-3000V, with a mean of 1800V.

**MTBF (MEAN TIME BEFORE FAILURE)**

**CONSIDERATIONS**

Results total: 212,000 device hours at 125°C, 0 failures  
301,000 device hours at 100°C, 0 failures

Assume: E<sub>a</sub> = 0.7 eV  
P<sub>d</sub> = 800 mW  
θ<sub>ja</sub> = 45°C/W

Chi-square statistics, 60% confidence  
Then: MTBF<sub>min</sub> at 25°C ambient = 93,000,000 device hours.  
MTBF<sub>min</sub> at 70°C ambient = 5,100,000 device hours.

# DP83910A CMOS SNI Serial Network Interface

## General Description

The DP83910A CMOS Serial Network Interface (SNI) is a direct-pin equivalent of the bipolar DP8391 SNI and provides the Manchester data encoding and decoding functions for IEEE 802.3 Ethernet/Thin-Ethernet type local area networks. The SNI interfaces the DP8390 Network Interface Controller (NIC) to the DP8392 CTI or an Ethernet transceiver cable. When transmitting, the SNI converts non-return-to-zero (NRZ) data from the controller into Manchester data and sends the converted data differentially to the transceiver. Conversely, when receiving, a Phase Lock Loop decodes the 10 Mbit/s data from the transceiver into NRZ data for the controller.

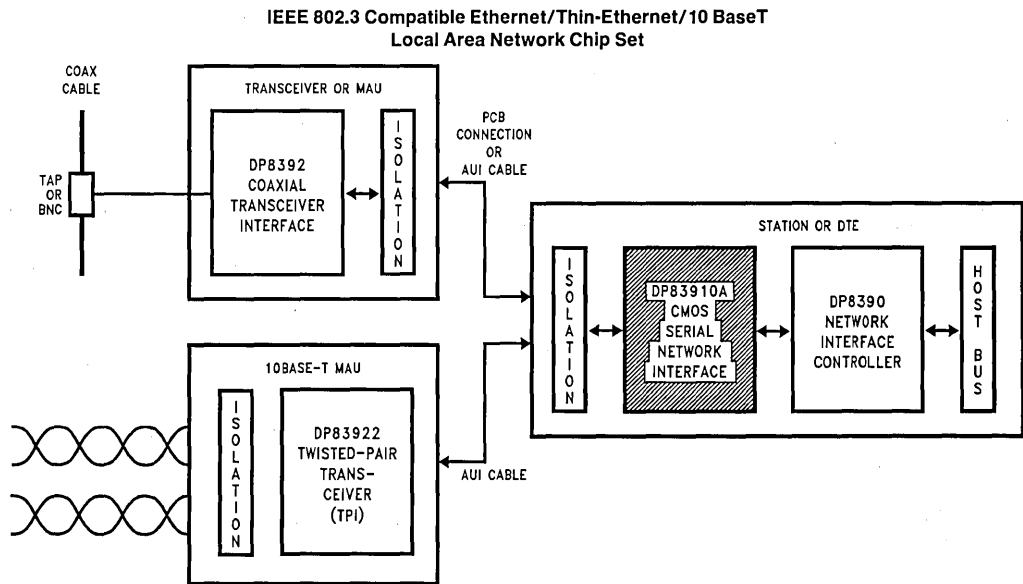
The DP83910A operates in conjunction with the DP8392 Coaxial Transceiver Interface (CTI) and the DP8390 Network Interface Controller (NIC) to form a three-chip set that implements a complete IEEE 802.3 compatible network as shown below. The DP83910A is a functionally complete Manchester encoder/decoder including a balanced driver and receiver, on-board crystal oscillator, collision signal translator, and a diagnostic loopback feature. The

DP83910A, fabricated CMOS, typically consumes less than 70 mA of current. However, as a result of being CMOS, the DP83910A's differential signals must be isolated in both Ethernet and thin wire Ethernet.

## Features

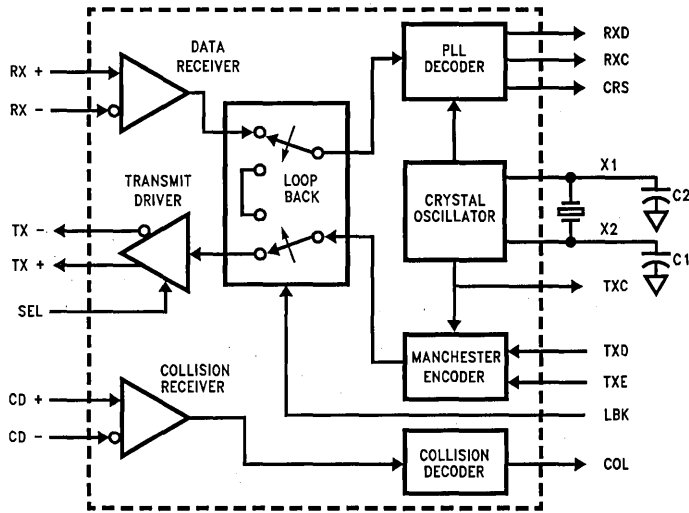
- Compatible with Ethernet I, IEEE 802.3; 10BASE5, 10BASE2, and 10BASE-T
- Designed to interface with 10BASE-T transceivers
- Functional and pin-out duplicate of the DP8391
- 10 Mbits/s Manchester encoding/decoding with receive clock recovery
- Requires no precision components
- Loopback capability for diagnostics
- Externally selectable half or full step modes of operation at transmit output
- Squelch circuitry at the receive and collision inputs to reject noise
- TTL/MOS compatible controller interface

## 1.0 System Diagram



TL/F/9365-1

## 2.0 Block Diagram



TL/F/9365-2

## 3.0 Functional Description

The DP83910A consists of five main logical blocks:

- The oscillator generates the 10 MHz transmit clock signal for system timing.
- The Manchester encoder accepts NRZ data from the controller, encodes the data to Manchester, and transmits it differentially to the transceiver, through the differential transmit driver.
- The Manchester decoder receives Manchester data from the transceiver, converts it to NRZ data and clock pulses, and sends it to the controller.
- The collision translator indicates to the controller the presence of a valid 10 MHz collision signal to the PLL.
- The loopback circuitry, when asserted, routes the data from the Manchester encoder back to the PLL decoder.

### 3.1 OSCILLATOR

The oscillator is controlled by a 20 MHz parallel resonant crystal connected between X1 and X2 or by an external clock on X1. The 20 MHz output of the oscillator is divided by 2 to generate the 10 MHz transmit clock for the controller. The oscillator also provides internal clock signals to the encoding and decoding circuits.

If a crystal is connected to the DP83910A, it is recommended that the circuit shown in *Figure 1* be used and that the components used meet the following:

Crystal XT1: AT cut parallel resonant crystal

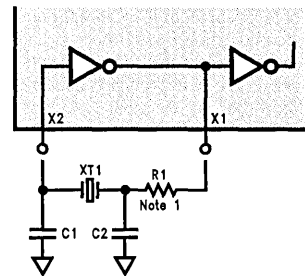
Series Resistance:  $\leq 10\Omega$

Specified Load Capacitance: 13.5 pF

Accuracy: 0.005% (50 ppm)

C1, C2: Load Capacitor, 27 pF.

The resistor, R1, in *Figure 1* may be required in order to minimize frequency drift due to changes in the  $V_{CC}$  supply voltage. If R1 is required, its value must be carefully selected. R1 decreases the loop gain. Thus, if R1 is made too large, the loop gain will be greatly reduced and the crystal will not oscillate. If R1 is made too small, normal variations in the  $V_{CC}$  may cause the oscillation frequency to drift out of specification. As the first rule of thumb, the value of R1



TL/F/9365-15

**Note 1:** The resistor R1 may be required in order to minimize frequency drift due to changes in the  $V_{CC}$ . See text description.

**FIGURE 1. Crystal Connection to DP83910A**

(see text for component values)

should be made equal to five times the motional resistance of the crystal.

The motional resistance of 20 MHz crystals is usually in the range of  $10\Omega$  to  $30\Omega$ . This implies that a reasonable value for R1 should be in the range of  $50\Omega$ – $150\Omega$ .

The decision of whether or not to include R1 should be based upon measured variations of crystal frequency as each of the circuit parameters is varied.

According to the IEEE 802.3 standard, the entire oscillator circuit (crystal and amplifier) must be accurate to 0.01%. When using a crystal, the X1 pin is not guaranteed to provide a TTL compatible logic output, and should not be used to drive external standard logic. If additional logic needs to be driven, then an external oscillator should be used, as described in the following.

### 3.2 OSCILLATOR MODULE OPERATION

If the designer wishes to use a crystal clock oscillator, one that provides the following should be employed:

- TTL or CMOS output with a 0.01% frequency tolerance
- 40%–60% duty cycle
- $\geq 2$  TTL load output drive ( $I_{OL} = 3.2$  mA)

### 3.0 Functional Description (Continued)

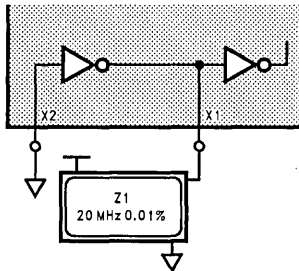
The circuit is shown in *Figure 2*. (Additional output drive may be necessary if the oscillator must also drive other components.) When using a clock oscillator it is still recommended that the designer connect the oscillator output to the X1 pin and tie the X2 pin to ground.

#### 3.3 MANCHESTER ENCODER AND DIFFERENTIAL DRIVER

The encoder begins operation when the Transmit Enable input (TXE) goes high and converts clock and NRZ data to Manchester data for the transceiver. For the duration of TXE remaining high, the Transmitted Data (TXD) is encoded for the transmit-driver pair (TX±). TXD must be valid on the rising edge of Transmit Clock (TXC). Transmission ends when TXE goes low. The last transition is always positive; it occurs at the center of the bit cell if the last bit is a one, or at the end of the bit cell if the last bit is a zero.

The differential transmit pair from the secondary of the isolation transformer drives up to 50 meters of twisted pair AUI cable. These outputs are source followers which require two 270Ω pull-down resistors to ground.

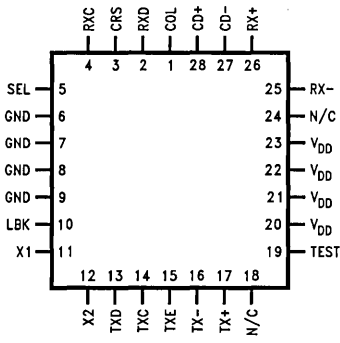
The DP83910A allows both half-step and full-step to be compatible with Ethernet I and IEEE 802.3. With the SEL pin low (for Ethernet I), transmit+ is positive with respect to transmit- during idle; with SEL high (for IEEE 802.3), transmit+ and transmit- are equal in the idle state. This provides zero differential voltage to operate with transformer coupled loads.



TL/F/9365-16

FIGURE 2. DP83910A Connection for Oscillator Module

### 4.0 Connection Diagrams



TL/F/9365-17

Top View

Order Number DP83910AV  
See NS Package Number V28A

#### 3.4 MANCHESTER DECODER

The decoder consists of a differential receiver and a PLL to separate Manchester encoded data stream into clock signals and NRZ data. The differential input must be externally terminated with two 39Ω resistors connected in series if the standard 78Ω transceiver drop cable is used; in Thin-Ethernet applications, these resistors are optional. To prevent noise from falsely triggering the decoder, a squelch circuit at the input rejects signals with levels less than -175 mV. Once the input exceeds the squelch requirements, Carrier Sense (CRS) is asserted. Receive data (RXD) and receive clock (RXC) become valid typically within 6 bit times. The DP83910A may tolerate bit jitter up to 18 ns in the received data.

The decoder detects the end of a frame when no more midbit transitions are detected. Within one and a half bit times after the last bit, carrier sense is de-asserted. Receive clock stays active for five more bit times after CRS goes low to guarantee the receive timings of the DP8390 NIC.

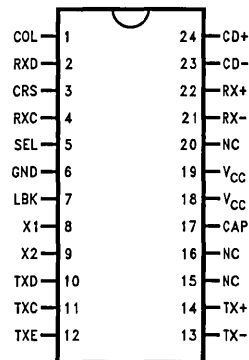
#### 3.5 COLLISION TRANSLATOR

When the Ethernet transceiver (DP8392 CTI) detects a collision, it generates a 10 MHz signal to the differential collision inputs (CD±) of the DP83910A. When these inputs are detected active, the DP83910A translates the 10 MHz signal to an active high level for the controller. The controller uses this signal to back off its current transmission and reschedule another one.

The collision differential inputs are terminated the same way as the differential receive inputs. The squelch circuitry is also similar, rejecting pulses with levels less than -175 mV.

#### 3.6 LOOPBACK FUNCTIONS

When the Loopback input (LBK) is asserted high, the DP83910A redirects its transmitted data back into its receive path. This feature provides a convenient method for testing both chip and system level integrity. The transmit driver and receive input circuitry are disabled in loopback mode.



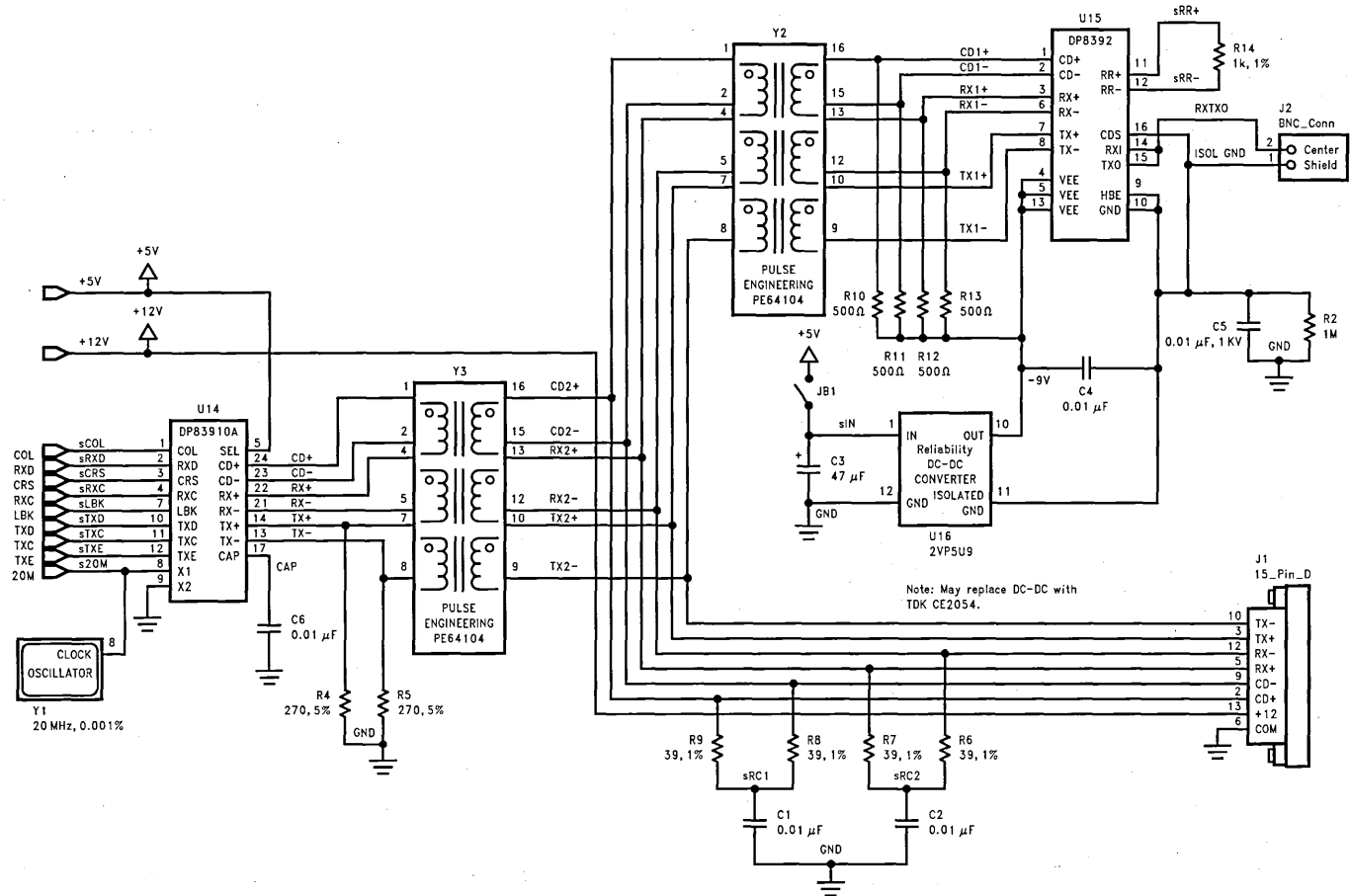
TL/F/9365-18

Top View

Order Number DP83910AN  
See NS Package Number N24C

Interface for Ethernet and Thin Wire Ethernet Using Single Jumper for Thin/Thick Selection

5.0 Typical Application



## 6.0 Pin Descriptions

| 24-Pin DIP | 28-Pin PCC           | Name                                                                     | I/O | Description                                                                                                                                                                                                    |
|------------|----------------------|--------------------------------------------------------------------------|-----|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1          | 1                    | COL                                                                      | O   | <b>COLLISION DETECT OUTPUT:</b> Generates an active high signal when 10 MHz collision signal is detected.                                                                                                      |
| 2          | 2                    | RXD                                                                      | O   | <b>RECEIVE DATA OUTPUT:</b> NRZ data output from the PLL. This signal must be sampled on the rising edge of receive clock.                                                                                     |
| 3          | 3                    | CRS                                                                      | O   | <b>CARRIER SENSE:</b> Asserted on the first valid high-to-low transition on the RX $\pm$ pair. Remains active until 1.5 bit times after the last bit in data.                                                  |
| 4          | 4                    | RXC                                                                      | O   | <b>RECEIVE CLOCK:</b> The receive clock from the Manchester data after the PLL has locked. Remains active 5 bit times after deasserting CRS.                                                                   |
| 5          | 5                    | SEL                                                                      | I   | <b>MODE SELECT:</b> When high, transmit+ and transmit- are the same voltage in the idle state. When low, transmit+ is positive with respect to transmit- in the idle state, at the transformer's primary.      |
| 6          | 7<br>8<br>9          | V <sub>SS</sub><br>V <sub>SS</sub><br>V <sub>SS</sub>                    |     | <b>GROUND PIN</b>                                                                                                                                                                                              |
| 7          | 10                   | LBK                                                                      | I   | <b>LOOPBACK:</b> When high, the loopback mode is enabled.                                                                                                                                                      |
| 8          | 11                   | X1                                                                       | I   | <b>CRYSTAL OR EXTERNAL OSCILLATOR INPUT</b>                                                                                                                                                                    |
| 9          | 12                   | X2                                                                       | O   | <b>CRYSTAL FEEDBACK OUTPUT:</b> Used in crystal connections only. Connected to ground when using an external oscillator.                                                                                       |
| 10         | 13                   | TXD                                                                      | I   | <b>TRANSMIT DATA INPUT:</b> NRZ data input from the controller. The data is combined with the transmit clock to produce Manchester data. TXD is sampled on the rising edge of transmit clock.                  |
| 11         | 14                   | TXC                                                                      | O   | <b>TRANSMIT CLOCK:</b> The 10 MHz clock derived from the 20 MHz oscillator.                                                                                                                                    |
| 12         | 15                   | TXE                                                                      | I   | <b>TRANSMIT ENABLE:</b> The encoder begins operation when this input is asserted high.                                                                                                                         |
| 13<br>14   | 16<br>17             | TX-<br>TX+                                                               | O   | <b>TRANSMIT OUTPUT:</b> Differential line driver which sends the encoded data to the transceiver. The outputs are source followers which require 270 $\Omega$ pull-down resistors.                             |
| 15         | 6                    | NC                                                                       |     | <b>NO CONNECTION:</b> This may be tied to V <sub>SS</sub> for the PLCC version to be compatible with the DP8391.                                                                                               |
| 16         | 18                   | NC                                                                       |     | <b>NO CONNECTION</b>                                                                                                                                                                                           |
| 17         | 19                   | TEST                                                                     | I   | <b>FACTORY TEST INPUT:</b> Used to check the chip's internal functions. May be tied low or have a 0.01 $\mu$ f bypass capacitor to ground (for compatibility with the bipolar DP8391) during normal operation. |
| 18<br>19   | 20<br>21<br>22<br>23 | V <sub>DD</sub><br>V <sub>DD</sub><br>V <sub>DD</sub><br>V <sub>DD</sub> |     | <b>POWER CONNECTION</b>                                                                                                                                                                                        |
| 20         | 24                   | NC                                                                       |     | <b>NO CONNECTION</b>                                                                                                                                                                                           |
| 21<br>22   | 25<br>26             | RX-<br>RX+                                                               | I   | <b>RECEIVE INPUT:</b> Differential receive input pair from the transceiver.                                                                                                                                    |
| 23<br>24   | 27<br>28             | CD-<br>CD+                                                               | I   | <b>COLLISION INPUT:</b> Differential collision pair input from the transceiver.                                                                                                                                |

## 7.0 Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

|                                 |                          |
|---------------------------------|--------------------------|
| Supply Voltage ( $V_{CC}$ )     | -0.5V to +7V             |
| DC Input Voltage ( $V_{IN}$ )   | -0.5V to $V_{CC} + 0.5V$ |
| DC Output Voltage ( $V_{OUT}$ ) | -0.5V to $V_{CC} + 0.5V$ |
| Differential Input Voltage      | -5.5 to +16V             |
| Differential Output Voltage     | 0 to 16V                 |
| Power Dissipation               | 500 mW                   |
| Storage Temperature             | -65°C to +150°C          |

|                                                                      |                     |
|----------------------------------------------------------------------|---------------------|
| Lead Temperature (Soldering, 10 sec.)                                | 260°C               |
| ESD ( $R_{ZAP} = 1.5 \text{ k}\Omega$ , $C_{ZAP} = 120 \text{ pF}$ ) | $\geq 2 \text{ kV}$ |
|                                                                      | (Pin 4 = 1.5 kV)    |

Note: Absolute maximum ratings are those values beyond which the safety of the device cannot be guaranteed. They are not meant to imply that the device should be operated at these limits.

\*Note: An asterisk following a parameter's symbol indicates that the parameter has been characterized but not tested.

Note: All specifications in this datasheet are valid only if the mandatory isolation is employed and all differential signals are taken to exist at the AUI side of the pulse transformer.

## 8.0 DC Specifications $T_A = 0^\circ\text{C}$ to $70^\circ\text{C}$ , $V_{CC} = 5V \pm 5\%$

| Symbol                                                                                    | Parameter                                                         | Conditions                                                               | Min       | Typ | Max            | Units         |
|-------------------------------------------------------------------------------------------|-------------------------------------------------------------------|--------------------------------------------------------------------------|-----------|-----|----------------|---------------|
| <b>Controller Interface Pins (COL, RXD, CRS, RXC, SEL, LBK, TXD, TXC and TXE)</b>         |                                                                   |                                                                          |           |     |                |               |
| $V_{IH}$                                                                                  | Input High Voltage                                                |                                                                          | 2.0       |     |                | V             |
| $V_{IL}$                                                                                  | Input Low Voltage                                                 |                                                                          |           |     | 0.8            | V             |
| $I_{IN}$                                                                                  | Input Leakage                                                     | $V_{IN} = V_{CC}$ or GND                                                 | -1.0      |     | 1.0            | $\mu\text{A}$ |
| $V_{OH}$                                                                                  | Output High Voltage                                               | (TTL) $I_{OH} = 2.0 \text{ mA}$<br>(CMOS) $I_{OH} = 20 \mu\text{A}$      | 3.5       |     | $V_{CC} - 0.1$ | V             |
| $V_{OL}$                                                                                  | Output Low Voltage                                                | (TTL) $I_{OL} = 2.0 \text{ mA}$<br>(CMOS) $I_{OL} = 20 \mu\text{A}$      |           |     | 0.4<br>0.1     | V             |
| $I_{CCO}$                                                                                 | Operating $V_{CC}$ Supply Current (Note 1)                        | 10 Mbit/sec                                                              |           |     | 70             | mA            |
| $I_{CCS}$                                                                                 | Stand By $V_{CC}$ Supply Current (Note 2)                         | 10 Mbit/sec                                                              |           |     | .65            | mA            |
| <b>Differential Pins (TX<math>\pm</math>, RX<math>\pm</math>, and CD<math>\pm</math>)</b> |                                                                   |                                                                          |           |     |                |               |
| $V_{OD}$                                                                                  | Diff. Output Voltage (TX $\pm$ )                                  | 78 $\Omega$ Termination, and<br>270 $\Omega$ from each to GND (Figure 4) | $\pm 550$ |     | $\pm 1200$     | mV            |
| $V_{OB}^*$                                                                                | Diff. Output Voltage Imbalance (TX $\pm$ )                        | 78 $\Omega$ Termination, and<br>270 $\Omega$ from each to GND (Figure 4) |           | 40  |                | mV            |
| $V_U^*$                                                                                   | Undershoot Voltage (TX $\pm$ )                                    | 78 $\Omega$ Termination, and<br>270 $\Omega$ from each to GND (Figure 4) |           | 100 |                | mV            |
| $V_{DS}$                                                                                  | Diff. Squelch Threshold (RX $\pm$ and CD $\pm$ )                  |                                                                          | -175      |     | -300           | mV            |
| $V_{CM}$                                                                                  | Diff. Input Common Mode Voltage (RX $\pm$ and CD $\pm$ ) (Note 3) |                                                                          | 0         |     | 5.5            | V             |
| <b>Oscillator Pins (X1 and X2)</b>                                                        |                                                                   |                                                                          |           |     |                |               |
| $V_{IH}$                                                                                  | X1 Input High Voltage                                             | X1 is connected to an oscillator, and X2 is grounded                     | 2.0       |     |                | V             |
| $V_{IL}$                                                                                  | X1 Input Low Voltage                                              | X1 is connected to an oscillator, and X2 is grounded                     |           |     | 0.8            | V             |
| $I_{OSC}$                                                                                 | X1 Input Current                                                  | X1 = $V_{CC}$ or GND<br>X2 = GND                                         | -2        |     | +2             | mA            |

**Note 1:** This measurement was made while the DP83910A was undergoing transmission, reception, and collision detection. Also, this value was not measured instantaneously, but averaged over a span of several milliseconds. ( $V_{IN} = 2.4V$  or  $0.4V$  and  $I_O = 0 \text{ mA}$ ).

**Note 2:** This measurement was made while the DP83910A was sitting idle with TXE low. Also, this value was not measured instantaneously, but averaged over a span of several milliseconds. ( $V_{IN} = 2.4V$  or  $0.4V$  and  $I_O = 0 \text{ mA}$ ).

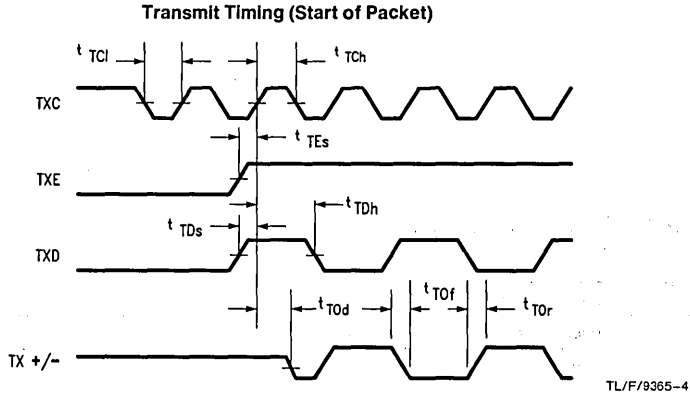
**Note 3:** This parameter is guaranteed by design and is not tested.



## 9.0 Switching Characteristics $T_A = 0^\circ\text{C to } 70^\circ\text{C}, V_{CC} = 5\text{V} \pm 5\%$

### Oscillator Specification

| Symbol    | Parameter                 | Min | Max | Units |
|-----------|---------------------------|-----|-----|-------|
| $t_{XTH}$ | X1 to Transmit Clock High | 5   | 30  | ns    |
| $t_{XTL}$ | X1 to Transmit Clock Low  | 5   | 30  | ns    |

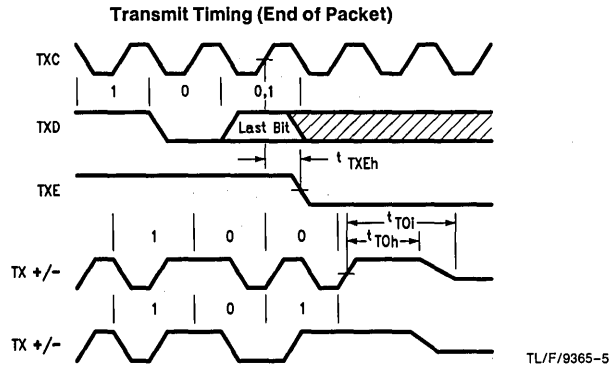


### Transmit Specifications (Start of Packet)

| Symbol      | Parameter                                                      | Min         | Max    | Units |
|-------------|----------------------------------------------------------------|-------------|--------|-------|
| $t_{TCh}$   | Transmit Clock High Time (Note 1)                              | 40          | 60     | ns    |
| $t_{TCI}$   | Transmit Clock Low Time (Note 1)                               | 40          | 60     | ns    |
| $t_{TCc}^*$ | Transmit Clock Cycle Time (Note 1)                             | 99.99       | 100.01 | ns    |
| $t_{TCr}^*$ | Transmit Clock Rise Time (20% to 80%) ( $C_L = 30\text{ pF}$ ) |             | 8      | ns    |
| $t_{TCf}^*$ | Transmit Clock Fall Time (80% to 20%) ( $C_L = 30\text{ pF}$ ) |             | 8      | ns    |
| $t_{TEs}$   | Transmit Enable Setup Time to Rising Edge of TXC (Note 1)      | 20          |        | ns    |
| $t_{TDs}$   | Transmit Data Setup Time from Rising Edge of TXC (Note 1)      | 20          |        | ns    |
| $t_{TDh}$   | Transmit Data Hold Time from Rising Edge of TXC                | 0           |        | ns    |
| $t_{TOd}$   | Transmit Output Delay from Rising Edge of TXC (Note 1)         |             | 65     | ns    |
| $t_{TOf}^*$ | Transmit Output Fall Time (80% to 20%)                         |             | 7      | ns    |
| $t_{TOr}^*$ | Transmit Output Rise Time (20% to 80%)                         |             | 7      | ns    |
| $t_{TOj}^*$ | Transmit Output Jitter                                         | 0.5 Typical |        | ns    |

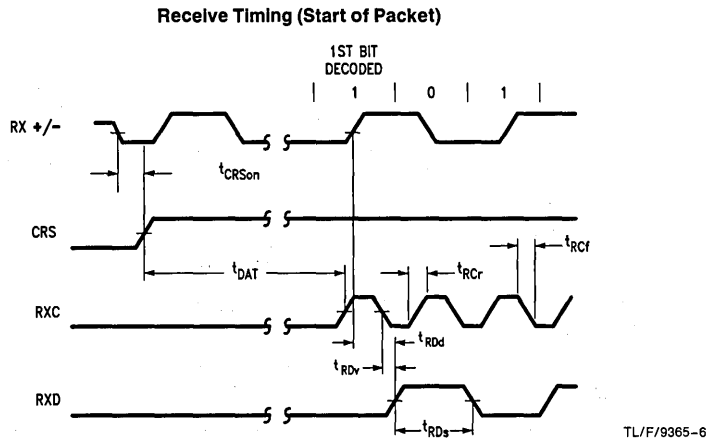
**Note 1:** This parameter is measured using the fifty percent point of each clock edge.

## 9.0 Switching Characteristics (Continued)



### Transmit Specifications (End of Packet)

| Symbol      | Parameter                                         | Min | Max  | Units |
|-------------|---------------------------------------------------|-----|------|-------|
| $t_{TXEh}$  | Transmit Enable Hold Time from Rising Edge of TXC | 0   |      | ns    |
| $t_{TOh}$   | Transmit Output High before Idle (Half Step)      | 200 |      | ns    |
| $t_{TOi}^*$ | Transmit Output Idle Time (Half Step)             |     | 8000 | ns    |



### Receiver Specifications (Start of Packet)

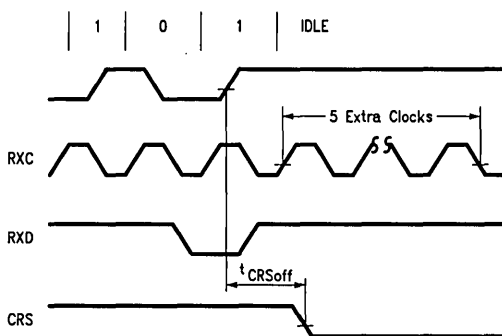
| Symbol      | Parameter                                              | Min | Max | Units |
|-------------|--------------------------------------------------------|-----|-----|-------|
| $t_{RCd}$   | Receive Clock Duty Cycle (Note 1)                      | 40  | 60  | %     |
| $t_{RCr}^*$ | Receive Clock Rise Time (20% to 80%, $C_{TL} = 30$ pF) |     | 7   | ns    |
| $t_{RCf}^*$ | Receive Clock Fall Time (80% to 20%, $C_{TL} = 30$ pF) |     | 7   | ns    |
| $t_{CRSon}$ | Carrier Sense Turn On Delay                            |     | 70  | ns    |
| $t_{DAT}$   | Decoder Acquisition Time                               |     | 700 | ns    |
| $t_{RDd}$   | Receive Data Output Delay                              |     | 150 | ns    |
| $t_{RDs}$   | Receive Data Output Stable after Going Valid           | 90  |     | ns    |
| $t_{Dtor}$  | Differential Inputs Turn-On Pulse (Note 2)             | 30  |     | ns    |
| $t_{RDV}$   | Receive Data Output Valid from Falling Edge of RXC     |     | 10  | ns    |

**Note 1:** This parameter is measured using the fifty percent point of each clock edge.

**Note 2:** This parameter was characterized with a differential input of  $-375$  mV on the receive pair inputs.

## 9.0 Switching Characteristics (Continued)

Receive Timing (End of Packet)



TL/F/9365-7

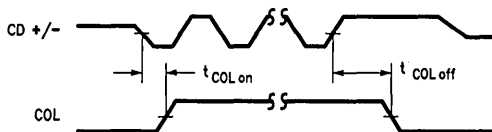
### Receiver Specifications (End of Packet)

| Symbol       | Parameter                                     | Min | Max | Units     |
|--------------|-----------------------------------------------|-----|-----|-----------|
| $t_{CRSoff}$ | Carrier Sense Turn Off Delay (Note 1)         |     | 155 | ns        |
| $t_{RXCh}$   | Minimum Number of RXCs after CRS Low (Note 2) | 5   |     | Bit Times |

Note 1: When CRS goes low, it will go low a minimum of 2 receive clocks.

Note 2: The DP8390 Network Interface Controller (NIC) requires a minimum of 5 receive clocks after CRS goes low to function properly.

Collision Timing



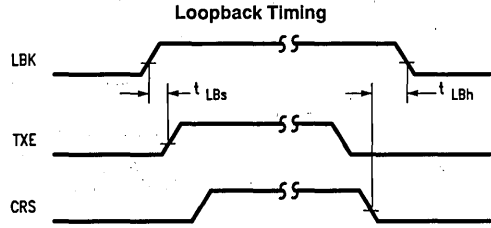
TL/F/9365-8

### Collision Specifications

| Symbol       | Parameter                                           | Min | Max | Units |
|--------------|-----------------------------------------------------|-----|-----|-------|
| $t_{COLon}$  | Collision Turn On Delay                             |     | 60  | ns    |
| $t_{COLoff}$ | Collision Turn Off Delay                            |     | 350 | ns    |
| $t_{Dtoc}$   | Differential Inputs Turn-On Pulse (Squelch, Note 1) | 30  |     | ns    |

Note 1: This parameter was characterized with a differential input of  $-375$  mV on the collision input pair.

## 9.0 Switching Characteristics (Continued)



TL/F/9365-9

### Loopback Specifications

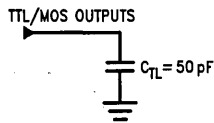
| Symbol    | Parameter                    | Min  | Max | Units |
|-----------|------------------------------|------|-----|-------|
| $t_{LBs}$ | Loopback Setup Time (Note 1) | 50   |     | ns    |
| $t_{LBh}$ | Loopback Hold Time (Note 1)  | 1000 |     | ns    |

**Note 1:** This parameter is guaranteed by design and is not tested.

### AC Timing Test Conditions

All specifications are valid only if the mandatory isolation is employed and all differential signals are taken to be at the AUI side of the pulse transformer.

|                                              |                               |
|----------------------------------------------|-------------------------------|
| Input Pulse Levels (TTL/CMOS)                | GND to 3.0V                   |
| Input Rise and Fall Times (TTL/CMOS)         | 5 ns                          |
| Input and Output Reference Levels (TTL/CMOS) | 1.3V                          |
| Input Pulse Levels (Diff.)                   | -350 to -1315 mV              |
| Input and Output Reference Levels (Diff.)    | 50% Point of the Differential |

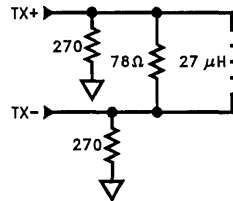


**FIGURE 3**

TL/F/9365-10

### Capacitance $T_A = 25^\circ\text{C}, f = 1\text{ MHz}$

| Symbol    | Parameter          | Typ | Units |
|-----------|--------------------|-----|-------|
| $C_{IN}$  | Input Capacitance  | 7   | pF    |
| $C_{OUT}$ | Output Capacitance | 7   | pF    |



TL/F/9365-12

**FIGURE 4**

**Note:** In the above diagram, the TX+ and TX- signals are taken from the AUI side of the isolation (pulse transformer). The pulse transformer used for all testing is the Pulse Engineering PE64103.

## DP8391A/NS32491A SNI Serial Network Interface

### General Description

The DP8391A Serial Network Interface (SNI) provides the Manchester data encoding and decoding functions for IEEE 802.3 Ethernet/Cheapernet type local area networks. The SNI interfaces the DP8390 Network Interface Controller (NIC) to the Ethernet transceiver cable. When transmitting, the SNI converts non-return-to-zero (NRZ) data from the controller and clock pulses into Manchester encoding and sends the converted data differentially to the transceiver. The opposite process occurs on the receive path, where a digital phase-locked loop decodes 10 Mbit/s signals with as much as  $\pm 18$  ns of jitter.

The DP8391A SNI is a functionally complete Manchester encoder/decoder including ECL like balanced driver and receivers, on board crystal oscillator, collision signal translator, and a diagnostic loopback circuit.

The SNI is part of a three chip set that implements the complete IEEE compatible network node electronics as shown below. The other two chips are the DP8392 Coax Transceiver Interface (CTI) and the DP8390 Network Interface Controller (NIC).

Incorporated into the CTI are the transceiver, collision and jabber functions. The Media Access Protocol and the buffer management tasks are performed by the NIC. There is an isolation requirement on signal and power lines between the CTI and the SNI. This is usually accomplished by using a set of miniature pulse transformers that come in a 16-pin plastic DIP for signal lines. Power isolation, however, is done by using a DC to DC converter.

### Features

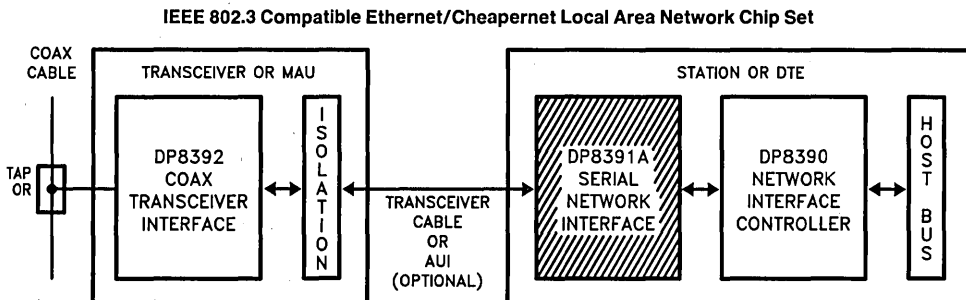
- Compatible with Ethernet II, IEEE 802.3; 10Base5, 10Base2, and 10Base-T

- 10 Mb/s Manchester encoding/decoding with receive clock recovery
- Patented digital phase locked loop (DPLL) decoder requires no precision external components
- Decodes Manchester data with up to  $\pm 18$  ns of jitter
- Loopback capability for diagnostics
- Externally selectable half or full step modes of operation at transmit output
- Squelch circuits at the receive and collision inputs reject noise
- High voltage protection at transceiver interface (16V)
- TTL/MOS compatible controller interface
- Connects directly to the transceiver (AUI) cable

### Table of Contents

- 1.0 System Diagram
- 2.0 Block Diagram
- 3.0 Functional Description
  - 3.1 Oscillator
  - 3.2 Encoder
  - 3.3 Decoder
  - 3.4 Collision Translator
  - 3.5 Loopback
- 4.0 Connection Diagrams
- 5.0 Pin Descriptions
- 6.0 Absolute Maximum Ratings
- 7.0 Electrical Characteristics
- 8.0 Switching Characteristics
- 9.0 Timing and Load Diagrams
- 10.0 Physical Dimensions

### 1.0 System Diagram



TL/F/9357-1

## 2.0 Block Diagram

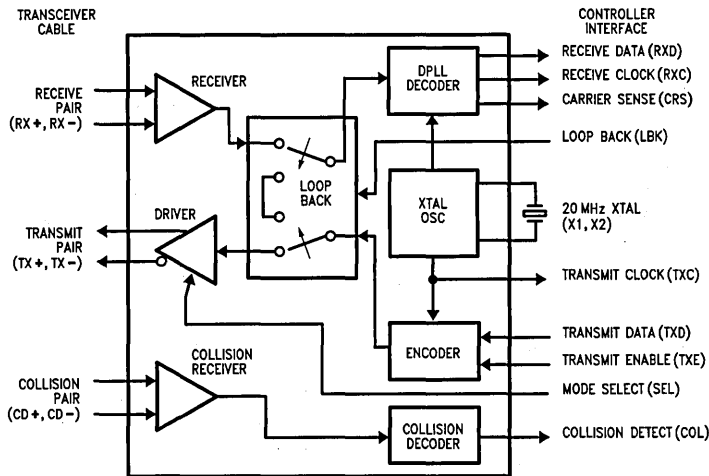


FIGURE 1

TL/F/9357-2

## 3.0 Functional Description

The SNI consists of five main logical blocks:

- the oscillator—generates the 10 MHz transmit clock signal for system timing.
- the Manchester encoder and differential output driver—accepts NRZ data from the controller, performs Manchester encoding, and transmits it differentially to the transceiver.
- the Manchester decoder—receives Manchester data from the transceiver, converts it to NRZ data and clock pulses, and sends them to the controller.
- the collision translator—indicates to the controller the presence of a valid 10 MHz signal at its input.
- the loopback circuitry—when asserted, switches encoded data instead of receive input signals to the digital phase-locked loop.

### 3.1 OSCILLATOR

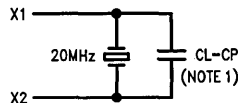
The oscillator is controlled by a 20 MHz parallel resonant crystal connected between X1 and X2 or by an external clock on X1. The 20 MHz output of the oscillator is divided by 2 to generate the 10 MHz transmit clock for the controller. The oscillator also provides internal clock signals to the encoding and decoding circuits.

#### Crystal Specification

|                    |                    |
|--------------------|--------------------|
| Resonant frequency | 20 MHz             |
| Tolerance          | ±0.001% at 25°C    |
| Stability          | ±0.005% 0–70°C     |
| Type               | AT-Cut             |
| Circuit            | Parallel Resonance |

The 20 MHz crystal connection to the SNI requires special care. The IEEE 802.3 standard requires a 0.01% absolute accuracy on the transmitted signal frequency. Stray capacitance can shift the crystal's frequency out of range, causing

the transmitted frequency to exceed its 0.01% tolerance. The frequency marked on the crystal is usually measured with a fixed shunt capacitance ( $C_L$ ) that is specified in the crystal's data sheet. This capacitance for 20 MHz crystals is typically 20 pF. The capacitance between the X1 and X2 pins of the SNI, of the PC board traces and the plated through holes plus any stray capacitance such as the socket capacitance, if one is used, should be estimated or measured. Once the total sum of these capacitances is determined, the value of additional external shunt capacitance required can be calculated. This capacitor can be a fixed 5% tolerance component. The frequency accuracy should be measured during the design phase at the transmit clock pin (TXC) for a given pc layout. *Figure 2* shows the crystal connection.



TL/F/9357-3

CL = Load capacitance specified by the crystal's manufacturer

CP = Total parasitic capacitance including:

- SNI input capacitance between X1 and X2 (typically 5 pF)
- PC board traces, plated through holes, socket capacitances

**Note 1:** When using a Viking (San Jose) VXB49N5 crystal, the external capacitor is not required, as the  $C_L$  of the crystal matches the input capacitance of the DP8391A.

### FIGURE 2. Crystal Connection

### 3.2 MANCHESTER ENCODER AND DIFFERENTIAL DRIVER

The encoder combines clock and data information for the transceiver. Data encoding and transmission begins with the transmit enable input (TXE) going high. As long as TXE is

### 3.0 Functional Description (Continued)

mains high, transmit data (TXD) is encoded out to the transmit-driver pair (TX±). The transmit enable and transmit data inputs must meet the setup and hold time requirements with respect to the rising edge of transmit clock. Transmission ends with the transmit enable input going low. The last transition is always positive at the transmit output pair. It will occur at the center of the bit cell if the last bit is one, or at the boundary of the bit cell if the last bit is zero.

The differential line driver provides ECL like signals to the transceiver with typically 5 ns rise and fall times. It can drive up to 50 meters of twisted pair AUI Ethernet transceiver cable. These outputs are source followers which need external 270Ω pulldown resistors to ground. Two different modes, full-step or half-step, can be selected with SEL input. With SEL low, transmit + is positive with respect to transmit - in the idle state. With SEL high, transmit + and transmit - are equal in the idle state, providing zero differential voltage to operate with transformer coupled loads. *Figures 4, 5 and 6* illustrate the transmit timing.

#### 3.3 MANCHESTER DECODER

The decoder consists of a differential input circuitry and a digital phase-locked loop to separate Manchester encoded data stream into clock signals and NRZ data. The differential input should be externally terminated if the standard 78Ω transceiver drop cable is used. Two 39Ω resistors connected in series and one optional common mode bypass capacitor would accomplish this. A squelch circuit at the input rejects signals with pulse widths less than 5 ns (negative going), or with levels less than -175 mV. Signals more negative than -300 mV and with a duration greater than 30 ns are always decoded. This prevents noise at the input from falsely triggering the decoder in the absence of a valid signal. Once the input exceeds the squelch requirements,

carrier sense (CRS) is asserted. Receive data (RXD) and receive clock (RXC) become available typically within 6 bit times. At this point the digital phase-locked loop has locked to the incoming signal. The DP8391A decodes a data frame with up to ±18 ns of jitter correctly.

The decoder detects the end of a frame when the normal mid-bit transition on the differential input ceases. Within one and a half bit times after the last bit, carrier sense is de-asserted. Receive clock stays active for five more bit times before it goes low and remains low until the next frame. *Figures 7, 8 and 9* illustrate the receive timing.

#### 3.4 COLLISION TRANSLATOR

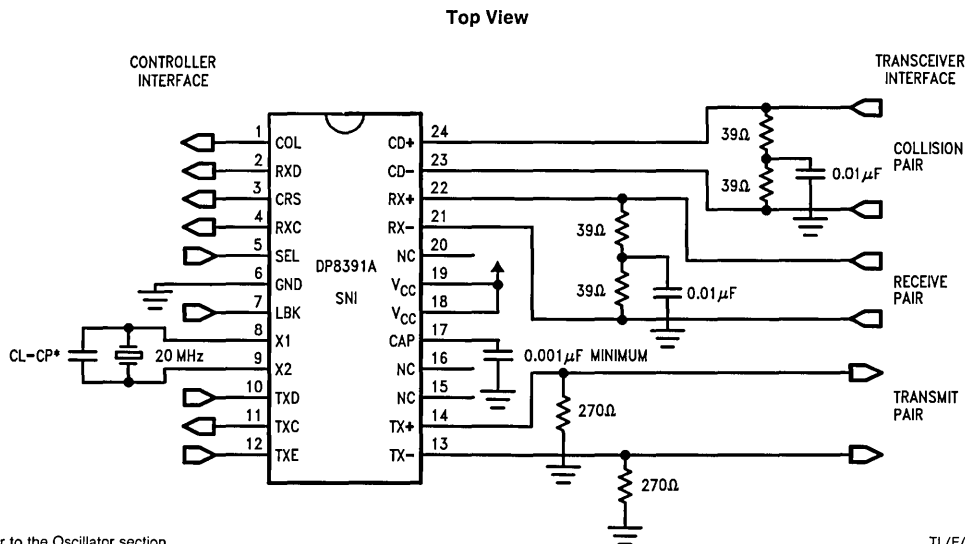
The Ethernet transceiver detects collisions on the coax cable and generates a 10 MHz signal on the transceiver cable. The SNI's collision translator asserts the collision detect output (COL) to the DP8390 controller when a 10 MHz signal is present at the collision inputs. The controller uses this signal to back off transmission and recycle itself. The collision detect output is de-asserted within 350 ns after the 10 MHz input signal disappears.

The collision differential inputs (+ and -) should be terminated in exactly the same way as the receive inputs. The collision input also has a squelch circuit that rejects signals with pulse widths less than 5 ns (negative going), or with levels less than -175 mV. *Figure 10* illustrates the collision timing.

#### 3.5 LOOPBACK FUNCTIONS

Logic high at loopback input (LBK) causes the SNI to route serial data from the transmit data input, through its encoder, returning it through the phase-locked-loop decoder to receive data output. In loopback mode, the transmit driver is in idle state and the receive and collision input circuitries are disabled.

### 4.0 Connection Diagram

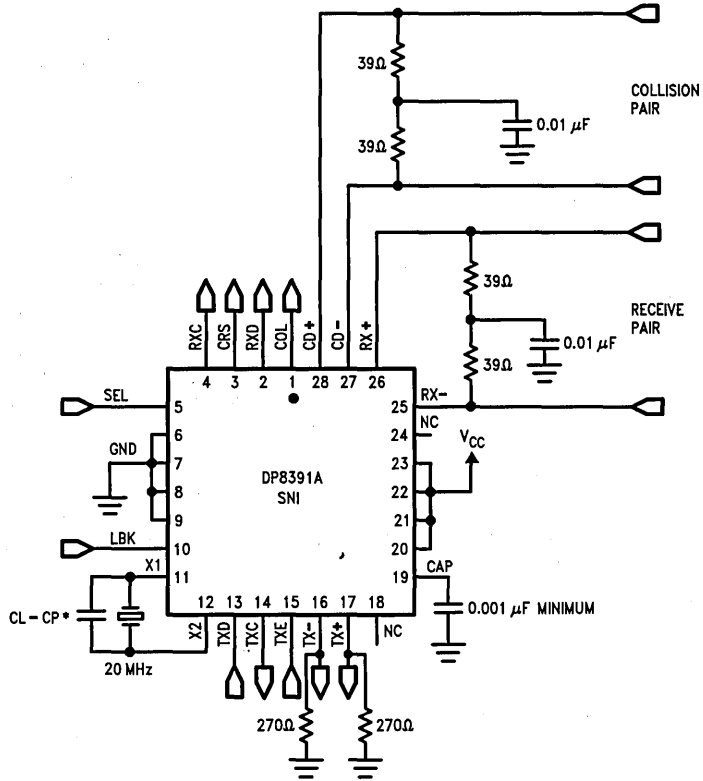


\*Refer to the Oscillator section

FIGURE 3a

Order Number DP8391AN  
See NS Package Number N24C

# PCC Connection Diagram



\*Refer to the Oscillator section

TL/F/9357-5

**FIGURE 3b**  
**Order Number DP8391AV**  
**NS Package Number V28A**



## 5.0 Pin Descriptions

| Pin No.  |          | Name       | I/O | Description                                                                                                                                                                                                                                                                                        |
|----------|----------|------------|-----|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| (DIP)    | (PCC)    |            |     |                                                                                                                                                                                                                                                                                                    |
| 1        | 1        | COL        | O   | <b>Collision Detect Output.</b> A TTL/MOS level active high output. A 10 MHz (+25%–15%) signal at the collision input will produce a logic high at COL output. When no signal is present at the collision input, COL output will go low.                                                           |
| 2        | 2        | RXD        | O   | <b>Receive Data Output.</b> A TTL/MOS level signal. This is the NRZ data output from the digital phase-locked loop. This signal should be sampled by the controller at the rising edge of receive clock.                                                                                           |
| 3        | 3        | CRS        | O   | <b>Carrier Sense.</b> A TTL/MOS level active high signal. It is asserted when valid data from the transceiver is present at the receive input. It is de-asserted one and a half bit times after the last bit at receive input.                                                                     |
| 4        | 4        | RXC        | O   | <b>Receive Clock.</b> A TTL/MOS level recovered clock. When the phase-locked loop locks to a valid incoming signal a 10 MHz clock signal is activated on this output. This output remains low during idle (5 bit times after activity ceases at receive input).                                    |
| 5        | 5        | SEL        | I   | <b>Mode Select.</b> A TTL level input. When high, transmit + and transmit – outputs are at the same voltage in idle state providing a “zero” differential. When low, transmit + is positive with respect to transmit – in idle state.                                                              |
| 6        | 6–9      | GND        |     | <b>Negative Supply Pins.</b>                                                                                                                                                                                                                                                                       |
| 7        | 10       | LBK        | I   | <b>Loopback.</b> A TTL level active high on this input enables the loopback mode.                                                                                                                                                                                                                  |
| 8        | 11       | X1         | I   | <b>Crystal or External Frequency Source Input (TTL).</b>                                                                                                                                                                                                                                           |
| 9        | 12       | X2         | O   | <b>Crystal Feedback Output.</b> This output is used in the crystal connection only. It must be left open when driving X1 with an external frequency source.                                                                                                                                        |
| 10       | 13       | TXD        | I   | <b>Transmit Data.</b> A TTL level input. This signal is sampled by the SNI at the rising edge of transmit clock when transmit enable input is high. The SNI combines transmit data and transmit clock signals into a Manchester encoded bit stream and sends it differentially to the transceiver. |
| 11       | 14       | TXC        | O   | <b>Transmit Clock.</b> A TTL/MOS level 10 MHz clock signal derived from the 20 MHz oscillator. This clock signal is always active.                                                                                                                                                                 |
| 12       | 15       | TXE        | I   | <b>Transmit Enable.</b> A TTL level active high data encoder enable input. This signal is also sampled by the SNI at the rising edge of transmit clock.                                                                                                                                            |
| 13<br>14 | 16<br>17 | TX–<br>TX+ | O   | <b>Transmit Output.</b> Differential line driver which sends the encoded data to the transceiver. These outputs are source followers and require 270Ω pulldown resistors to GND.                                                                                                                   |
| 15<br>16 | 18       | NC         |     | <b>No Connection.</b>                                                                                                                                                                                                                                                                              |
| 17       | 19       | CAP        | O   | <b>Bypass Capacitor.</b> A ceramic capacitor (greater than 0.001 μF) must be connected from this pin to GND.                                                                                                                                                                                       |
| 18<br>19 | 20–23    | VCC        |     | <b>Positive Supply Pins.</b> A 0.1 μF ceramic decoupling capacitor must be connected across VCC and GND as close to the device as possible.                                                                                                                                                        |
| 20       | 24       | NC         |     | <b>No Connection.</b>                                                                                                                                                                                                                                                                              |
| 21<br>22 | 25<br>26 | RX–<br>RX+ | I   | <b>Receive Input.</b> Differential receive input pair from the transceiver.                                                                                                                                                                                                                        |
| 23<br>24 | 27<br>28 | CD–<br>CD+ | I   | <b>Collision Input.</b> Differential collision input pair from the transceiver.                                                                                                                                                                                                                    |

## 6.0 Absolute Maximum Ratings

|                                                            |               |
|------------------------------------------------------------|---------------|
| Supply Voltage ( $V_{CC}$ )                                | 7V            |
| Input Voltage (TTL)                                        | 0 to 5.5V     |
| Input Voltage (differential)                               | -5.5 to +16V  |
| Output Voltage (differential)                              | 0 to 16V      |
| Output Current (differential)                              | -40 mA        |
| Storage Temperature                                        | -65° to 150°C |
| Lead Temperature (soldering, 10 sec)                       | 300°C         |
| Package Power Rating for DIP at 25°C<br>(PC Board Mounted) | 2.95W*        |
| Derate Linearly at the rate of 23.8 mW/°C                  |               |
| Package Power Rating for PCC at 25°C                       | 1.92W*        |
| Derate Linearly at the rate of 15.4 mW/°C                  |               |

\*For actual power dissipation of the device please refer to Section 7.0.

ESD rating 2000V

## Recommended Operating Conditions

|                             |            |
|-----------------------------|------------|
| Supply Voltage ( $V_{CC}$ ) | 5V ± 5%    |
| Ambient Temperature (DIP)   | 0° to 70°C |
| (PCC)                       | 0° to 55°C |

Note: Absolute maximum ratings are those values beyond which the safety of the device cannot be guaranteed. They are not meant to imply that the device should be operated at these limits.

## 7.0 Electrical Characteristics

$V_{CC} = 5V \pm 5\%$ ,  $T_A = 0^\circ C$  to  $70^\circ C$  for DIP and  $0^\circ C$  to  $55^\circ C$  for PCC (Notes 1 & 2)

| Symbol      | Parameter                                                  | Test Conditions                                               | Min   | Max            | Units              |
|-------------|------------------------------------------------------------|---------------------------------------------------------------|-------|----------------|--------------------|
| $V_{IH}$    | Input High Voltage (TTL)                                   |                                                               | 2.0   |                | V                  |
| $V_{IHx1a}$ | Input High Voltage (X1)                                    | No Series Resistor                                            | 2.0   | $V_{CC} - 1.5$ | V                  |
| $V_{IHx1b}$ | Input High Voltage (X1)                                    | 1k Series Resistor                                            | 2.0   | $V_{CC}$       | V                  |
| $V_{IL}$    | Input Low Voltage (TTL and X1)                             |                                                               |       | 0.8            | V                  |
| $I_{IH}$    | Input High Current (TTL)<br>Input High Current (RX ± CD ±) | $V_{IN} = V_{CC}$<br>$V_{IN} = V_{CC}$                        |       | 50<br>500      | $\mu A$<br>$\mu A$ |
| $I_{IL}$    | Input Low Current (TTL)<br>Input Low Current (RX ± CD ±)   | $V_{IN} = 0.5V$<br>$V_{IN} = 0.5V$                            |       | -300<br>-700   | $\mu A$<br>$\mu A$ |
| $V_{CL}$    | Input Clamp Voltage (TTL)                                  | $I_{IN} = -12 mA$                                             |       | -1.2           | V                  |
| $V_{OH}$    | Output High Voltage (TTL/MOS)                              | $I_{OH} = -100 \mu A$                                         | 3.5   |                | V                  |
| $V_{OL}$    | Output Low Voltage (TTL/MOS)                               | $I_{OL} = 8 mA$                                               |       | 0.5            | V                  |
| $I_{OS}$    | Output Short Circuit Current (TTL/MOS)                     |                                                               | -40   | -200           | mA                 |
| $V_{OD}$    | Differential Output Voltage (TX ±)                         | 78 $\Omega$ termination, and<br>270 $\Omega$ from each to GND | ± 550 | ± 1200         | mV                 |
| $V_{OB}$    | Diff. Output Voltage Imbalance (TX ±)                      | same as above                                                 |       | ± 40           | mV                 |
| $V_{DS}$    | Diff. Squelch Threshold (RX ± CD ±)                        |                                                               | -175  | -300           | mV                 |
| $V_{CM}$    | Diff. Input Common Mode Voltage (RX ± CD ±)                |                                                               | -5.25 | 5.25           | V                  |
| $I_{CC}$    | Power Supply Current                                       | 10Mbit/s                                                      |       | 270            | mA                 |

## 8.0 Switching Characteristics $V_{CC} = 5V \pm 5\%$ , $T_A = 0^\circ C$ to $70^\circ C$ for DIP and $0^\circ C$ to $55^\circ C$ for PCC (Note 2)

| Symbol                          | Parameter                                               | Figure | Min | Typ | Max | Units |
|---------------------------------|---------------------------------------------------------|--------|-----|-----|-----|-------|
| <b>OSCILLATOR SPECIFICATION</b> |                                                         |        |     |     |     |       |
| $t_{XTH}$                       | X1 to Transmit Clock High                               | 12     | 8   |     | 20  | ns    |
| $t_{XTL}$                       | X1 to Transmit Clock Low                                | 12     | 8   |     | 20  | ns    |
| <b>TRANSMIT SPECIFICATION</b>   |                                                         |        |     |     |     |       |
| $t_{TCd}$                       | Transmit Clock Duty Cycle at 50% (10 MHz)               | 12     | 42  | 50  | 58  | %     |
| $t_{TCr}$                       | Transmit Clock Rise Time (20% to 80%)                   | 12     |     |     | 8   | ns    |
| $t_{TCf}$                       | Transmit Clock Fall Time (80% to 20%)                   | 12     |     |     | 8   | ns    |
| $t_{TDs}$                       | Transmit Data Setup Time to Transmit Clock Rising Edge  | 4 & 12 | 20  |     |     | ns    |
| $t_{TDh}$                       | Transmit Data Hold Time from Transmit Clock Rising Edge | 4 & 12 | 0   |     |     | ns    |
| $t_{TEs}$                       | Transmit Enable Setup Time to Trans. Clock Rising Edge  | 4 & 12 | 20  |     |     | ns    |
| $t_{TEh}$                       | Transmit Enable Hold Time from Trans. Clock Rising Edge | 5 & 12 | 0   |     |     | ns    |

Note 1: All currents into device pins are positive, all currents out of device pins are negative. All voltages are referenced to ground unless otherwise specified.

Note 2: All typicals are given for  $V_{CC} = 5V$  and  $T_A = 25^\circ C$ .

## 8.0 Switching Characteristics

$V_{CC} = 5V \pm 5\%$ ,  $T_A = 0^\circ\text{C}$  to  $70^\circ\text{C}$  for DIP and  $0^\circ\text{C}$  to  $55^\circ\text{C}$  for PCC (Note 2) (Continued)

| Symbol                                    | Parameter                                             | Figure    | Min      | Typ        | Max  | Units         |
|-------------------------------------------|-------------------------------------------------------|-----------|----------|------------|------|---------------|
| <b>TRANSMIT SPECIFICATION (Continued)</b> |                                                       |           |          |            |      |               |
| $t_{TOd}$                                 | Transmit Output Delay from Transmit Clock Rising Edge | 4 & 12    |          |            | 50   | ns            |
| $t_{TOr}$                                 | Transmit Output Rise Time (20% to 80%)                | 12        |          |            | 7    | ns            |
| $t_{TOf}$                                 | Transmit Output Fall Time (80% to 20%)                | 12        |          |            | 7    | ns            |
| $t_{TOj}$                                 | Transmit Output Jitter                                | 12        |          | $\pm 0.25$ |      | ns            |
| $t_{TOh}$                                 | Transmit Output High Before Idle in Half Step Mode    | 5 & 12    | 200      |            |      | ns            |
| $t_{TOi}$                                 | Transmit Output Idle Time in Half Step Mode           | 5 & 12    |          |            | 800  | ns            |
| <b>RECEIVE SPECIFICATION</b>              |                                                       |           |          |            |      |               |
| $t_{RCd}$                                 | Receive Clock Duty Cycle at 50% (10 MHz)              | 12        | 40       | 50         | 60   | %             |
| $t_{RCr}$                                 | Receive Clock Rise Time (20% to 80%)                  | 12        |          |            | 8    | ns            |
| $t_{RCf}$                                 | Receive Clock Fall Time (80% to 20%)                  | 12        |          |            | 8    | ns            |
| $t_{RDr}$                                 | Receive Data Rise Time (20% to 80%)                   | 12        |          |            | 8    | ns            |
| $t_{RDf}$                                 | Receive Data Fall Time (80% to 20%)                   | 12        |          |            | 8    | ns            |
| $t_{RDs}$                                 | Receive Data Stable from Receive Clock Rising Edge    | 7 & 12    | $\pm 40$ |            |      | ns            |
| $t_{CSon}$                                | Carrier Sense Turn On Delay                           | 7 & 12    |          |            | 50   | ns            |
| $t_{CSoff}$                               | Carrier Sense Turn Off Delay                          | 8, 9 & 12 |          |            | 160  | ns            |
| $t_{DAT}$                                 | Decoder Acquisition Time                              | 7         |          | 0.6        | 1.80 | $\mu\text{s}$ |
| $t_{Drej}$                                | Differential Inputs Rejection Pulse Width (Squelch)   | 7         | 5        |            | 30   | ns            |
| $t_{Rd}$                                  | Receive Throughput Delay                              | 8 & 12    |          |            | 150  | ns            |
| <b>COLLISION SPECIFICATION</b>            |                                                       |           |          |            |      |               |
| $t_{COLon}$                               | Collision Turn On Delay                               | 10 & 12   |          |            | 50   | ns            |
| $t_{COLoff}$                              | Collision Turn Off Delay                              | 10 & 12   |          |            | 350  | ns            |
| <b>LOOPBACK SPECIFICATION</b>             |                                                       |           |          |            |      |               |
| $t_{LBs}$                                 | Loopback Setup Time                                   | 11        | 20       |            |      | ns            |
| $t_{LBh}$                                 | Loopback Hold Time                                    | 11        | 0        |            |      | ns            |

Note 2: All typicals are given for  $V_{CC} = 5V$  and  $T_A = 25^\circ\text{C}$ .

## 9.0 Timing and Load Diagrams

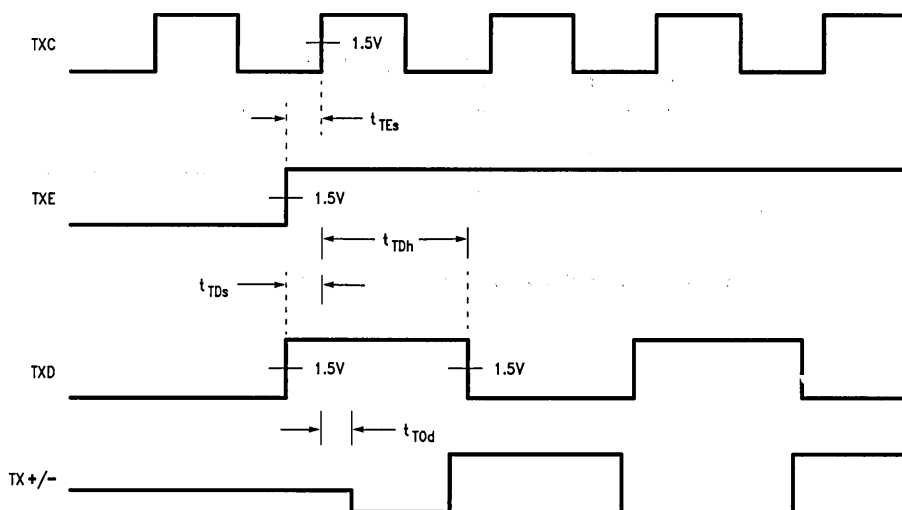


FIGURE 4. Transmit Timing - Start of Transmission

TL/F/9357-6

### 9.0 Timing and Load Diagrams (Continued)

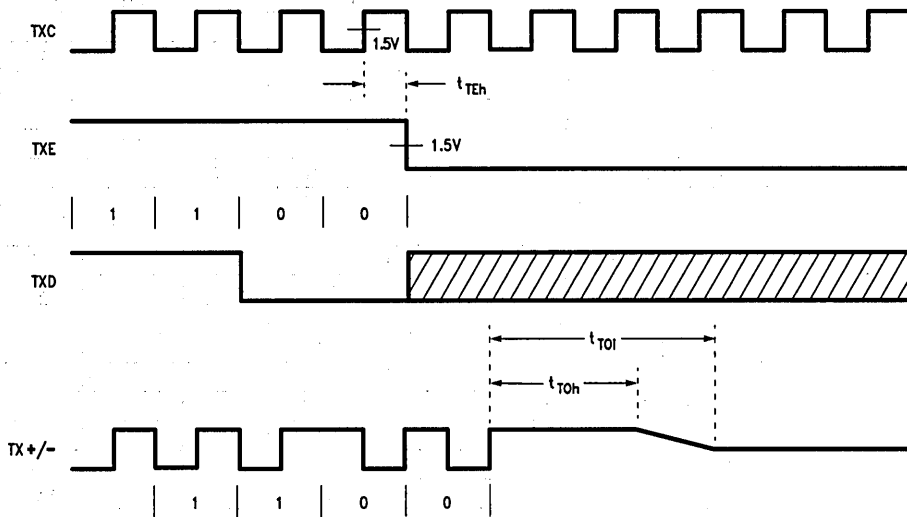


FIGURE 5. Transmit Timing - End of Transmission (last bit = 0)

TL/F/9357-7

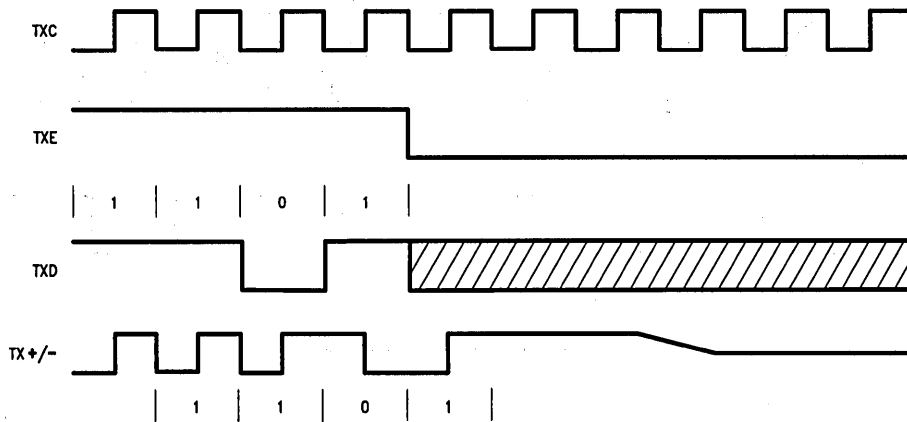


FIGURE 6. Transmit Timing - End of Transmission (last bit = 1)

TL/F/9357-8

9.0 Timing and Load Diagrams (Continued)

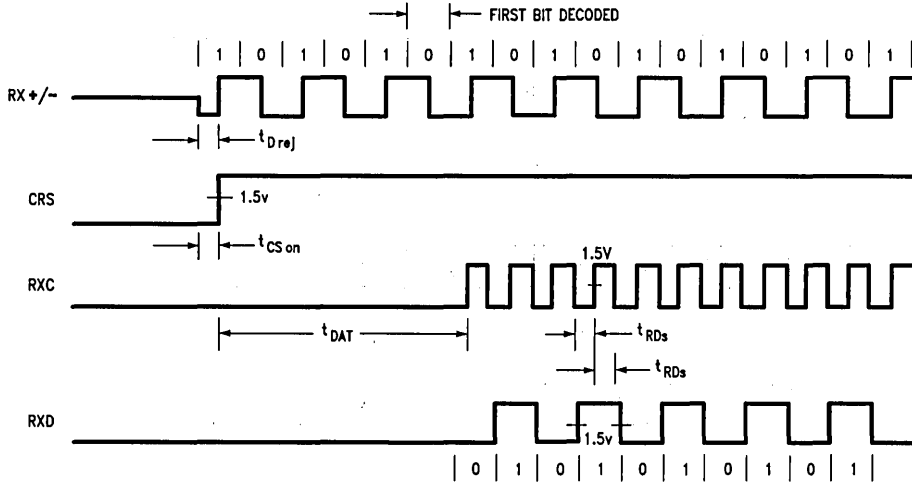


FIGURE 7. Receive Timing - Start of Packet

TL/F/9357-9

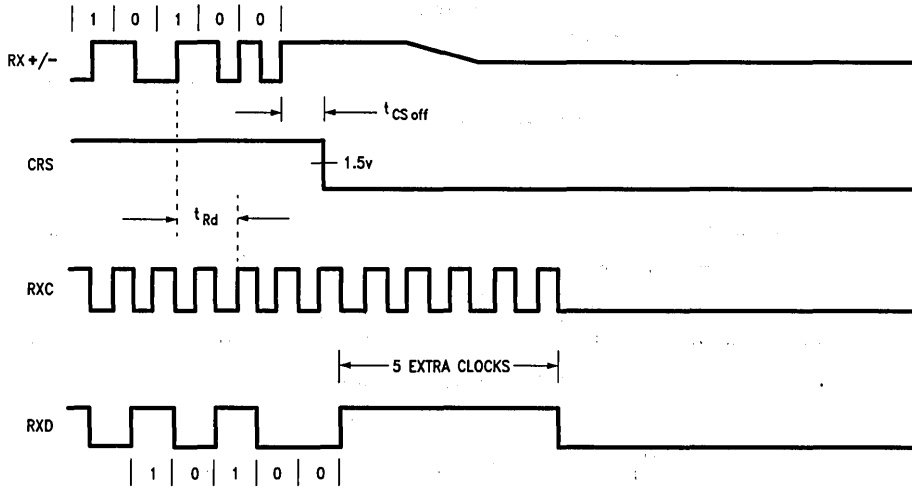


FIGURE 8. Receive Timing - End of Packet (last bit = 0)

TL/F/9357-10

### 9.0 Timing and Load Diagrams (Continued)

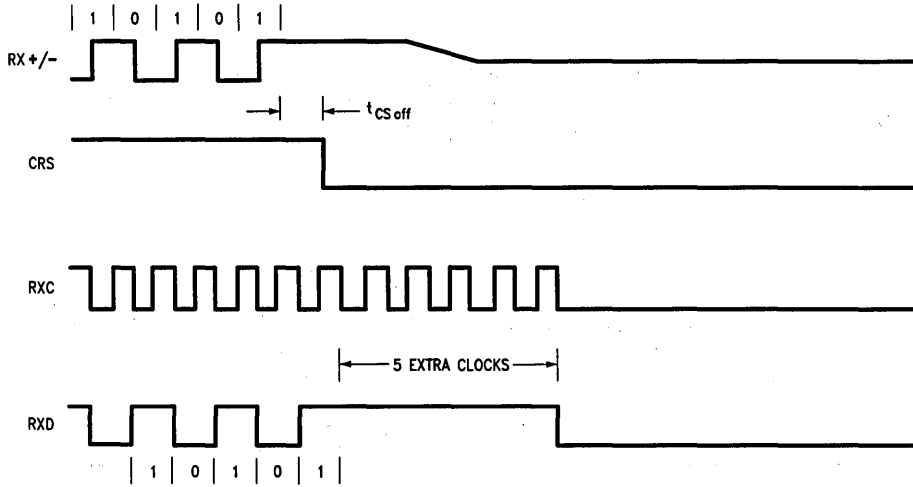


FIGURE 9. Receive Timing - End of Packet (last bit = 1)

TL/F/9357-11

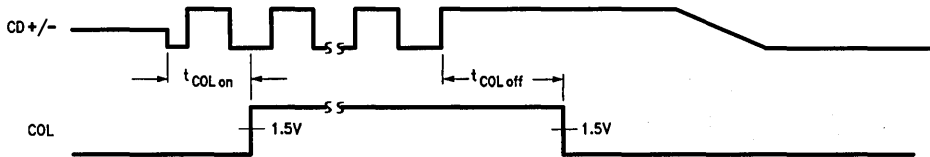


FIGURE 10. Collision Timing

TL/F/9357-12

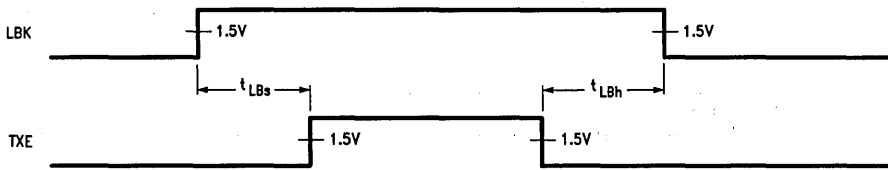
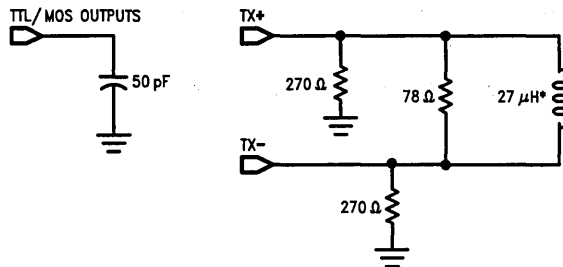


FIGURE 11. Loopback Timing

TL/F/9357-13



TL/F/9357-14

\*27 μH transformer is used for testing purposes, 100 μH transformers (Valor, LT1101, or Pulse Engineering 64103) are recommended for application use.

FIGURE 12. Test Loads

# Ethernet/Cheapernet Physical Layer Made Easy with DP8391/92

National Semiconductor  
Application Note 442  
Alex Djenguerian



AN-442

With the integration of the node electronics of IEEE 802.3 compatible local area networks now on silicon, system design is simplified. This application note describes the differences between the Ethernet and Cheapernet versions of the standard, and provides design guidelines for implementing the node electronics with National Semiconductor's DP8390 LAN chip set.

## INTRODUCTION

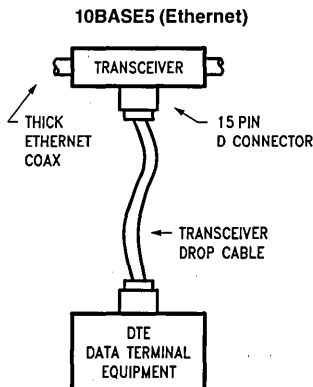
The DP8390 chip set is designed to provide the physical and media access control layer functions of local area networks as specified in IEEE 802.3 standard. This standard is based on the access method known as carrier-sense multiple access with collision detection (CSMA/CD). In this scheme, if a network station wants to transmit, it first "lis-

tens" to the medium; if someone else is transmitting, the station defers until the medium is clear before it begins to transmit. However, two or more stations could still begin transmitting at the same time and give rise to a collision. When this happens, the two nodes detect this condition, back off for a random amount of time before making another attempt.

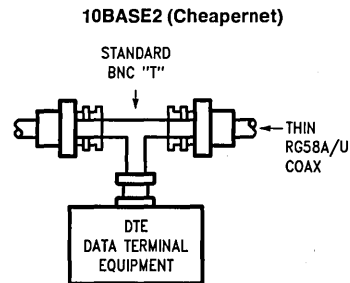
The IEEE 802.3 standard supports two different versions for the media, 10BASE5 (commonly known as Ethernet) and 10BASE2 (Cheapernet). These can be used separately, or together in a hybrid form. Both versions have similar electrical specifications and can be implemented using the same transceiver chip (DP8392). Cheapernet is the low cost version and is user installable. The following table compares the two:

| Parameter              | 10BASE5 (Ethernet)                                                          | 10BASE2 (Cheapernet)                                                              |
|------------------------|-----------------------------------------------------------------------------|-----------------------------------------------------------------------------------|
| Data Rate              | 10 Mbit/s baseband                                                          | 10 Mbps/s baseband                                                                |
| Segment Length         | 500 m                                                                       | 185 m                                                                             |
| Network Span           | 2500 m                                                                      | 925 m                                                                             |
| Nodes per Segment      | 100                                                                         | 30                                                                                |
| Node Spacing           | 2.5 m (cable marked)                                                        | 0.5 m min                                                                         |
| Capacitance per Node   | 4 pF max                                                                    | 8 pF max                                                                          |
| Cable                  | 0.4 in diameter<br>50Ω<br>Double Shielded<br>Rugged<br>N-Series Connectors  | 0.2 in diameter<br>50Ω (RG58A/U)<br>Single Shielded<br>Flexible<br>BNC Connectors |
| Transceiver Drop Cable | 0.39 in diameter multiway cable with<br>15 pin D connectors 50 m max length | Not needed due to the high flexibility of<br>the RG58A/U cable                    |

## Typical Connection Diagram for a Station



TL/F/8689-1



TL/F/8689-2

Although Cheapernet is intended for local use, several 185 meter segments can be joined together with simple repeaters to provide for a larger network span. Similarly, several Cheapernet segments can be tied into a longer Ethernet "backbone". In this hybrid configuration, the network com-

bins all the benefits of Cheapernet, flexibility and low cost, with the ruggedness and the much larger geographic range of standard Ethernet. Figure 1 illustrates a typical hybrid LAN configuration.

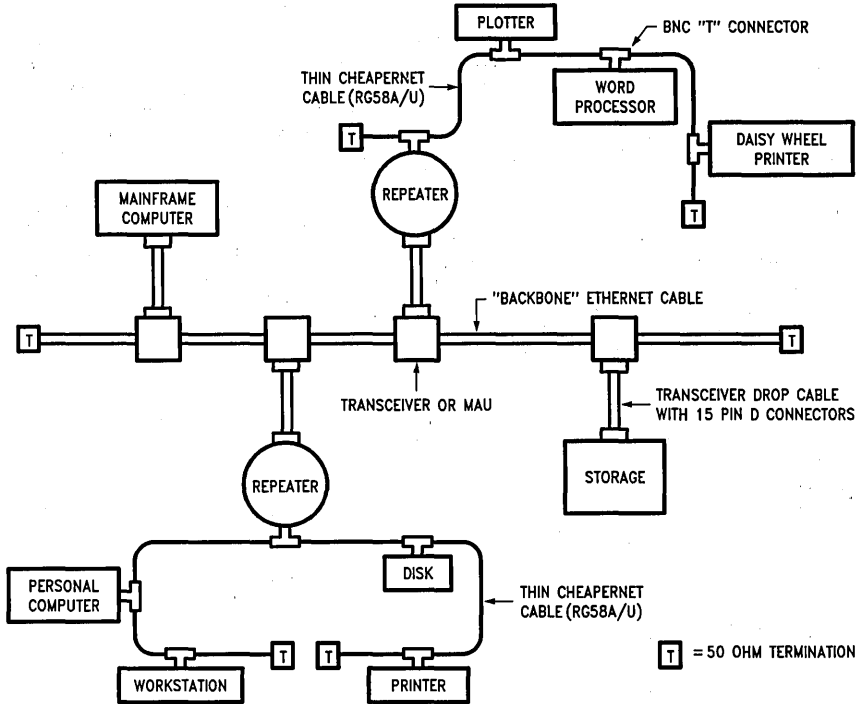
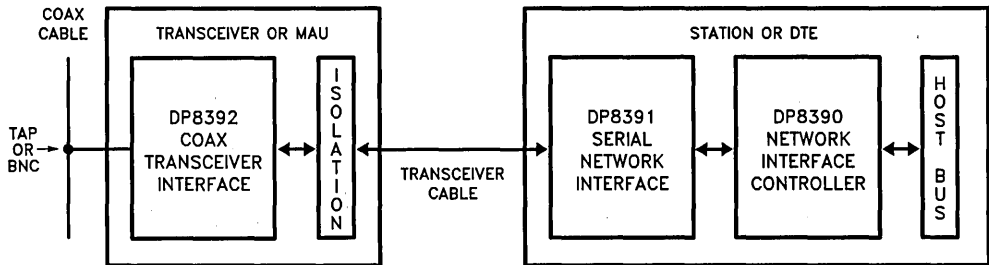


FIGURE 1. A Hybrid Ethernet/Cheapernet System

TL/F/8689-3

TRANSMITTING AND RECEIVING PACKETS WITH THE DP8390 CHIPSET

Node Block Diagram



TL/F/8689-4

The node electronics is integrated into three chips, the DP8390 Network Interface Controller (NIC), the DP8391 Serial Network Interface (SNI), and the DP8392 Coaxial Transceiver Interface (CTI). To transmit a packet, the host processor issues a transmit command to the NIC, which normal-

ly transfers the data to a local buffer memory. The NIC then automatically handles the transmission of the packet (from the local buffer through an on-board FIFO to the SNI) according to the CSMA/CD protocol. The packet has to be in the following format:

| PREAMBLE | SFD    | DESTINATION | SOURCE  | LENGTH  | DATA          | CRC     |
|----------|--------|-------------|---------|---------|---------------|---------|
| 62-bits  | 2-bits | 6-bytes     | 6-bytes | 2-bytes | 46-1500 bytes | 4-bytes |



**PREAMBLE:** This section consists of alternating 1 and 0 bits. As the packet travels through the network, some of these bits would be lost as most of the network components are allowed to provide an output some number of bits after being presented with a valid input.

**START OF A FRAME DELIMITER (SFD):** This field consists of two consecutive 1's to signal that the frame reception should begin.

**DESTINATION AND SOURCE ADDRESSES:** Each one of these frames is 6 bytes long and specifies the address of the corresponding node.

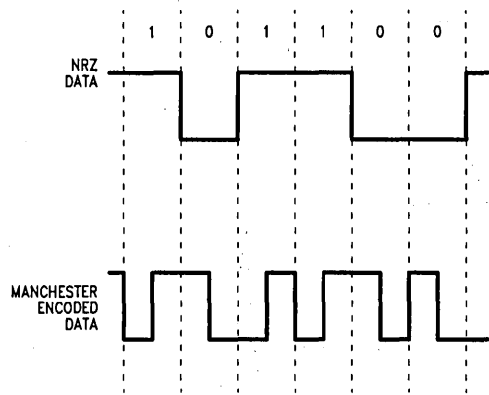
**LENGTH:** This 2 byte field indicates the number of bytes in the data field.

**DATA:** This field can be from 46 to 1500 bytes long. Messages shorter than 46 bytes require padding to bring the data field to the minimum length. If the data field is padded, the host can determine the number of valid data bytes by looking at the length field. Messages longer than 1500 bytes must be broken into multiple packets.

**CRC:** This field contains a Cyclic Redundancy Code calculation performed on the Destination address through the Data field for error control.

The shortest packet length thus adds up to be 512 bits long (excluding the preamble and the SFD). At 10 Mbit/sec this amounts to 51.2  $\mu$ s, which is twice as much as the 25  $\mu$ s maximum end-to-delay time that is allowed by the IEEE 802.3 protocol. This ensures that if a collision arises in the network, it would be recognized at all node locations.

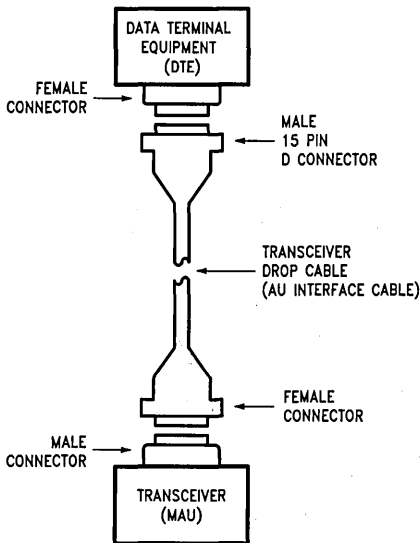
The SNI combines the NRZ data packet received from the controller with a clock signal and encodes them into a serial bit stream using standard Manchester encoding. In this coding scheme, the first half of the bit cell contains the complementary data and the second half contains the true data. Thus a transition is always guaranteed in the middle of a bit cell.



TL/F/8689-5

**FIGURE 2. Manchester Coding**

The encoded signal appears in differential form at the SNI's output. In 10BASE5 (Ethernet) applications, this signal is sent to the transceiver or the Medium Attachment Unit (MAU) through the twisted pair Transceiver Drop cable (also known as the Attachment Unit Interface cable). This cable typically consists of four individually shielded twisted wire pairs with an overall shield covering these individually shielded pairs. The signal pairs, which have a differential characteristic impedance of  $78\Omega \pm 5\Omega$ , should be terminated at the receiving ends. The cable can be up to 50 meters in length and have a maximum delay of 257 ns. The shields of the individual pairs should be connected to the logic ground in the Data Terminal Equipment (DTE) and the outer shield to the chassis ground. Figure 3 shows a picture of the cable and the corresponding pin assignments.



TL/F/8689-6

**FIGURE 3. Transceiver Cable Pin Assignments**

| Pin   | IEEE 802.3 Name      | Pairs          | DP8391/2 Name | Signal from |     |
|-------|----------------------|----------------|---------------|-------------|-----|
|       |                      |                |               | DTE         | MAU |
| 3     | DO + (Data Out +)    | Transmit Pair  | TX+           | X           |     |
| 10    | DO - (Data Out -)    |                | TX-           | X           |     |
| 11    | DO S (DO Shield)     |                |               | X           |     |
| 5     | DI + (Data In +)     | Receive Pair   | RX+           |             | X   |
| 12    | DI - (Data In -)     |                | RX-           |             | X   |
| 4     | DI S (DI Shield)     |                |               | X           |     |
| 7     | CO + (Control Out +) | Optional Pair  |               | X           |     |
| 15    | CO - (Control Out -) |                |               | X           |     |
| 8     | CO S (CO Shield)     |                |               | X           |     |
| 2     | CI + (Control In +)  | Collision Pair | CD+           |             | X   |
| 9     | CI - (Control In -)  |                | CD-           |             | X   |
| 1     | CI S (CI Shield)     |                |               | X           |     |
| 6     | VC (Voltage Common)  | Power Pair     |               | X           |     |
| 13    | VP (Voltage Plus)    |                |               | X           |     |
| 14    | VS (Voltage Shield)  |                |               | X           |     |
| Shell | PG (Protective GND)  |                |               | X           |     |

The transmitted packet from the SNI as well as all other signals (receive, collision, and DC power) must be electrically isolated from the coax in the MAU. The isolation means provided must withstand 500 V<sub>AC</sub> rms for one minute for 10BASE2 and 2000 V<sub>AC</sub> rms for 10BASE5. In order to detect collisions reliably, the electrical isolation is not done at the coax; instead it is done on the side of the Attachment Unit Interface. The isolation for the three signal lines can be easily provided by using three pulse transformers that come in a standard 16 pin plastic DIP from several manufacturers (Pulse Engineering, Valor Electronics). The inductance value for these transformers vary from 50  $\mu$ H to 150  $\mu$ H with the larger inductance values slowing the rise and fall times, and the smaller ones causing more voltage droop.

The Manchester encoded data from the SNI now reaches the CTI's transmit input after passing through the isolation transformer. A noise filter at this input provides a static noise margin of -175 mV to -300 mV. These thresholds assure that differential Transmit (TX $\pm$ ) data signals less than -175 mV or narrower than 10 ns are always rejected, while signals greater than -300 mV and wider than 30 ns are always accepted. The -300 mV threshold provides sufficient margin since the differential drivers for the transceiver drop cable provide a minimum signal level of  $\pm 450$  mV after inductive droop, and the maximum attenuation allowed for the drop cable is 3 dB at signal frequencies. Signals meeting the squelch requirements are waveshaped and outputted to the coax medium. This is done as follows:

The transmitter's output driver is a switching current source that drives a purely resistive load of 25 $\Omega$  presented by the coax to produce a voltage swing of approximately 2V. This

signal has to meet several critical electrical requirements:

**RISE/FALL TIMES:** The 10%–90% rise and fall times have to be 25 ns  $\pm$  5 ns at 10 Mbit/sec. This spec helps to minimize electro-magnetic radiation by reducing the higher harmonic content of the signal and contributes to the smaller reflection levels on the coax. In addition, the rise and fall times are required to be matched to within 1 ns to minimize the overall jitter in the system.

**DC LEVEL:** The DC component of the signal has to be between -37 mA and -45 mA. The tolerance here is tight since collisions are detected by monitoring the average DC level on the coax.

**AC LEVEL:** The AC component of the signal has to be between  $\pm 28$  mA and the DC level. This specification guarantees a minimum signal at the far end of the coax cable in the worst case condition.

The signal shown in Fig. 4 would be attenuated as it travels along the coax. The maximum cable attenuation per segment is 8.5 dB at 10 MHz and 6 dB at 5 MHz. This applies for both the 500 meters of Ethernet cable and the 185 meters of Cheapernet cable. With 10 Mbit/sec Manchester data, this cable attenuation results in approximately 7 ns of edge jitter in either direction. The CTI's receiver has to compensate for at least a portion of this jitter to meet the  $\pm 6$  ns combined jitter budget. The receiver also should not over-compensate the signal in the case of a short cable. An equalizer filter in the CTI accomplishes this task. *Figure 5* shows a typical waveform seen at the far end of the cable and the corresponding differential output from the CTI's receiver.

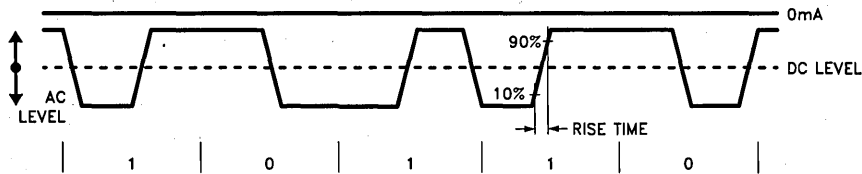


FIGURE 4. Coax Transmit Waveform

TL/F/8689-7

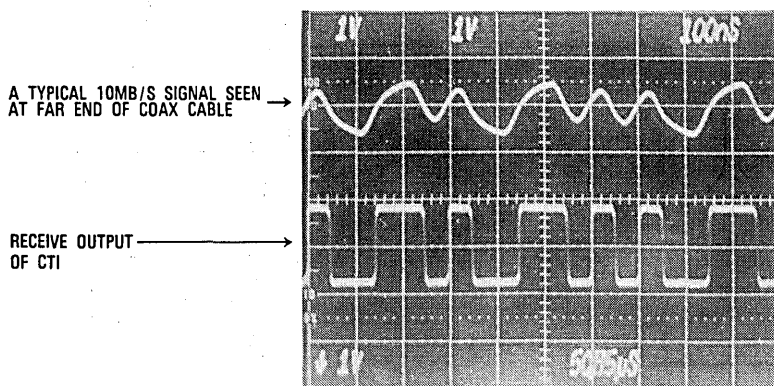
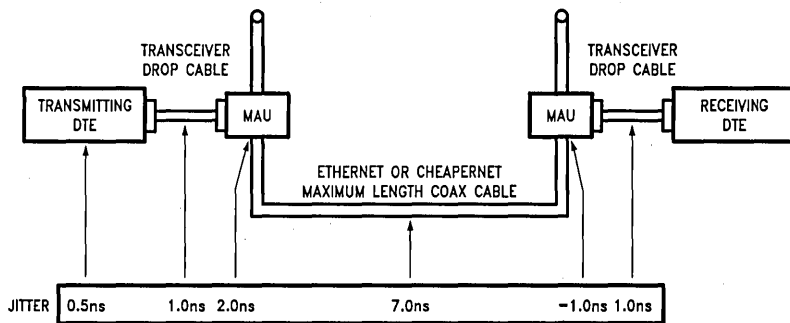


FIGURE 5. Oscilloscope Waveforms

TL/F/8689-8



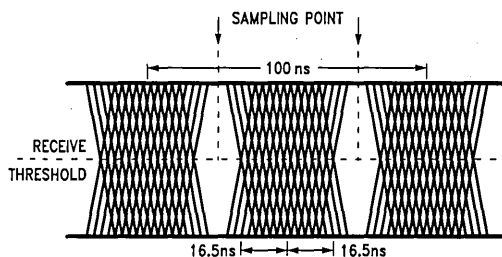
TL/F/8689-9

Total Jitter without Noise =  $0.5 + 1.0 + 2.0 + 7.0 - 1.0 + 1.0 = 10.5$  ns

Additional Jitter from Noise on Coax Cable = 5.0 ns

Additional Jitter from Noise on Drop Cables = 1.0 ns

Total System Jitter = 16.5 ns



TL/F/8689-10

FIGURE 6. Typical Signal Waveform at SNI's Input

In addition to the equalizer, an AC/DC squelch circuit at the coax input prevents noise on the cable from falsely triggering the receiver in the absence of a valid signal. The Receive differential line from the CTI should be isolated before it reaches the SNI for Manchester decoding. This signal now could have accumulated as much as  $\pm 16.5$  ns of jitter. Figure 6 illustrates the jitter allocations for different network components and a typical signal waveform at the SNI's input. The digital phase-locked loop of the SNI can decode Manchester data with up to  $\pm 20$  ns of random jitter which provides enough margin for implementation.

The SNI converts the Manchester received packet to NRZ data and clock pulses and sends them to the controller. Upon reception, the NIC checks the destination address, and if it is valid, verifies the CRC with the one generated on board and stores the packet in the local buffer memory. The packet is then moved to the host by the NIC, and when this is completed the buffer area is reclaimed for storing new packets. If a collision occurs during this transfer process, the CTI will detect it by sensing the average DC level on the coax and will send a 10 MHz collision signal to the SNI. The SNI will translate this information to the controller in TTL form, and the transmitting controllers will backoff for different times and retransmit later. Also in case of illegally long packets (longer than 20 ms), a jabber timer in the CTI will disable the coax driver so that the "jabbering" station will

not bring down the entire network. The collision pair is activated in this case to inform the controller of the faulty condition. After the fault is removed, the jabber timer holds for 500 ms before re-enabling the coax driver.

#### COLLISION DETECTION SCHEMES

There are two different collision detection schemes that can be implemented with the CTI; receive, transmit modes. The IEEE 802.3 standard allows the use of receive, transmit, and transhybrid modes for non-repeater nodes for both Ethernet and Cheapernet applications. Repeaters are required to have the receive mode implementation.

**RECEIVE MODE:** Detects a collision between any two stations on the network with certainty at all times.

**TRANSMIT MODE:** Detects collisions with certainty only when the station is transmitting.

**RECEIVE MODE:** The receive mode scheme has a very simple truth table; however, the tight threshold limits make the design of it difficult. The threshold in this case has to be between the maximum DC level of one station ( $-1300$  mV) and the minimum DC level of two far end stations ( $-1581$  mV). Several factors such as the termination resistor variation, coax center conductor resistance, driver current level variation, signal skew, and input bias current of non-transmitting nodes contribute to this tight margin. On

**Truth Table for Various  
Collision Detection Schemes**

| Mode             | Receive         |   |   |   | Transmit |   |   |   |    |
|------------------|-----------------|---|---|---|----------|---|---|---|----|
|                  | No. of Stations | 0 | 1 | 2 | >2       | 0 | 1 | 2 | >2 |
| Transmitting     | N               | N | Y | Y | N        | N | Y | Y |    |
| Non-Transmitting | N               | N | Y | Y | N        | N | M | Y |    |

Y = It will detect a collision, N = It will not detect a collision,  
M = It may detect a collision

top of the  $-1300$  mV minimum level, the impulse response of the internal low pass filter has to be added. The CTI incorporates a 4 pole Bessel filter in combination with a trimmed on board bandgap reference to provide this mode of collision detection. However it would be difficult in receive mode to extend the cable length beyond the limits of the standard. It is also argued that it is not necessary for non-repeater nodes to detect collisions between other stations.

**TRANSMIT MODE:** In this case collisions have to be detected with certainty only when the station is transmitting. Thus, collisions caused by two other nodes may or may not be detected. This feature relaxes the upper limit of the threshold from  $-1581$  mV to  $-1782$  mV. As a result of this, longer cable segments can be used. With the CTI, a resistor divider can be used at the Collision Detect Sense pin (CDS) to lower the threshold from receive to transmit mode. Typical resistor values can be  $120\Omega$  from CDS to GND and  $10k$  from CDS to  $V_{EE}$  (This moves the threshold by about  $-100$  mV).

#### IMPLEMENTING A 10 BASE5 (ETHERNET) MAU WITH THE DP8392

The CTI provides all the MAU (transceiver) functions except for signal and power isolation. Signal isolation can easily be provided by a set of three pulse transformers that come in a single Dual-in-Line package. These are available from transformer vendors such as Pulse Engineering (PE64103) and Valor (LT1101). However, for the power isolation a DC to DC converter is required. The CTI requires a single  $-9$  ( $\pm 5\%$ ) volt supply. This power has to be derived from the power pair of the drop cable which is capable of providing  $500$  mA in the  $12$  ( $-6\%$ ) to  $15$  ( $+5\%$ ) volt range. The low supply current of the CTI makes the design of the DC to DC converter quite easy. Such converters are being developed in hybrid packages by transformer manufacturers (Pulse Engineering PE64430 and Reliability Inc. 2E12R9). They provide the necessary voltage isolation and the output regulation. One can also build a simple DC to DC converter with a two transistor self oscillating primary circuit and some regulation on the secondary as shown in *Figure 7*.

Several areas of the PC board layout require special care. The most critical of these is for the coax connection. Ethernet requires that the CTI capacitance be less than  $2$  pF on the coax with another  $2$  pF allocated for the tap mechanism. The Receive Input (RXI) and the Transmit Output (TXO) lines should be kept to an absolute minimum by mounting the CTI very close to the center pin of the tap. Also, for the external diode at TXO (see *Figure 8*), the designer must minimize any stray capacitance, particularly on the anode side of the diode. To do this, all metal lines, especially the ground and  $V_{EE}$  planes, should be kept as far as possible from the RXI and TXO lines.

In order to meet the stringent capacitive loading requirements on the coax, it is imperative that the CTI be directly soldered to the PC board without a socket. A special lead frame in the CTI package allows direct conduction of heat from the die through these leads to the PC board, thus reducing the operating die temperature significantly. For good heat conduction the  $V_{EE}$  pins (4, 5 and 13) should be connected to large metal traces or planes.

A separate voltage sense pin (CDS) is provided for accurate detection of collision levels on the coax. In receive mode, where the threshold margin is tight, this pin should be independently attached to the coax shield to minimize errors due to ground drops. A resistor divider network at this pin can be used for transmit mode operation as described earlier.

The differential transmit pair from the DTE should be terminated with a  $78\Omega$  differential resistive load. By splitting the termination resistor into two equal values and capacitively AC grounding the center node, the common mode impedance is reduced to about  $20\Omega$ , which helps to attenuate common mode transients.

To drive the  $78\Omega$  differential line with sufficient voltage swings, the CTI's collision and receive drivers need external  $500\Omega$  resistors to  $V_{EE}$ . By using external resistors, the power dissipation of the chip is reduced, enhancing long term reliability. The only precision component required for the CTI is one  $1k$   $1\%$  resistor. This resistor sets many important parameters of the chip such as the coax driving levels, output rise and fall times,  $10$  MHz collision oscillator frequency, jabber timing, and receiver AC squelch timing. It should be connected between pins 11 (RR+) and 12 (RR-).

The DP8392 features a heartbeat function which can be externally disabled using pin 9. This function activates the collision output for a short time ( $10 \pm 5$  bit cells) at the end of every transmission. It is used to ensure the controller that the collision circuitry is intact and properly functioning. Pin 9 enables CD Heartbeat when grounded, and disables it when connected to  $V_{EE}$ .

The IEEE 802.3 standard requires a static discharge path to be provided between the shield of the coax cable and the DTE ground via a 1 MΩ, 0.25W resistor. The standard also requires the MAU to have low susceptibility levels to electromagnetic interference. A 0.01 μF capacitor will provide a

sufficient AC discharge path from the coaxial cable shield to the DTE ground. The individual shields should also be capacitively coupled to the Voltage Common in MAU. A typical Ethernet MAU connection diagram using the CTI in receive mode with the CD Heartbeat enabled is shown in Figure 8.

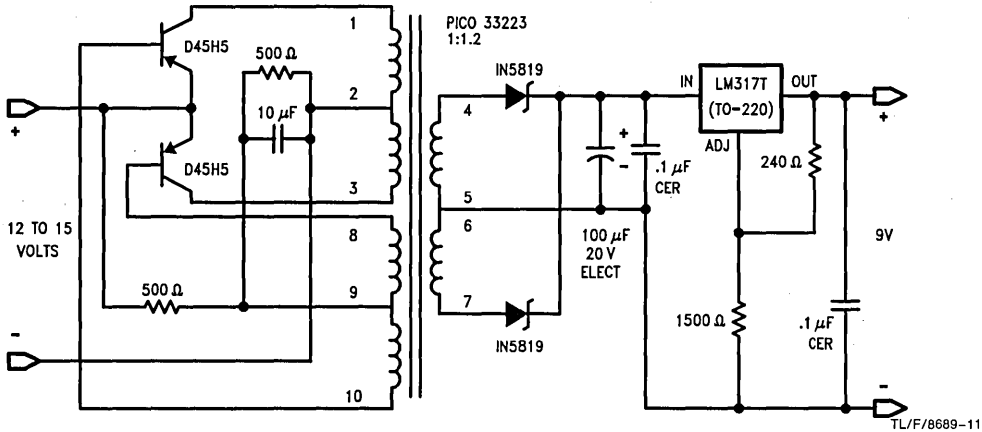


FIGURE 7. A Simple Low Cost DC to DC Converter

TL/F/8689-11

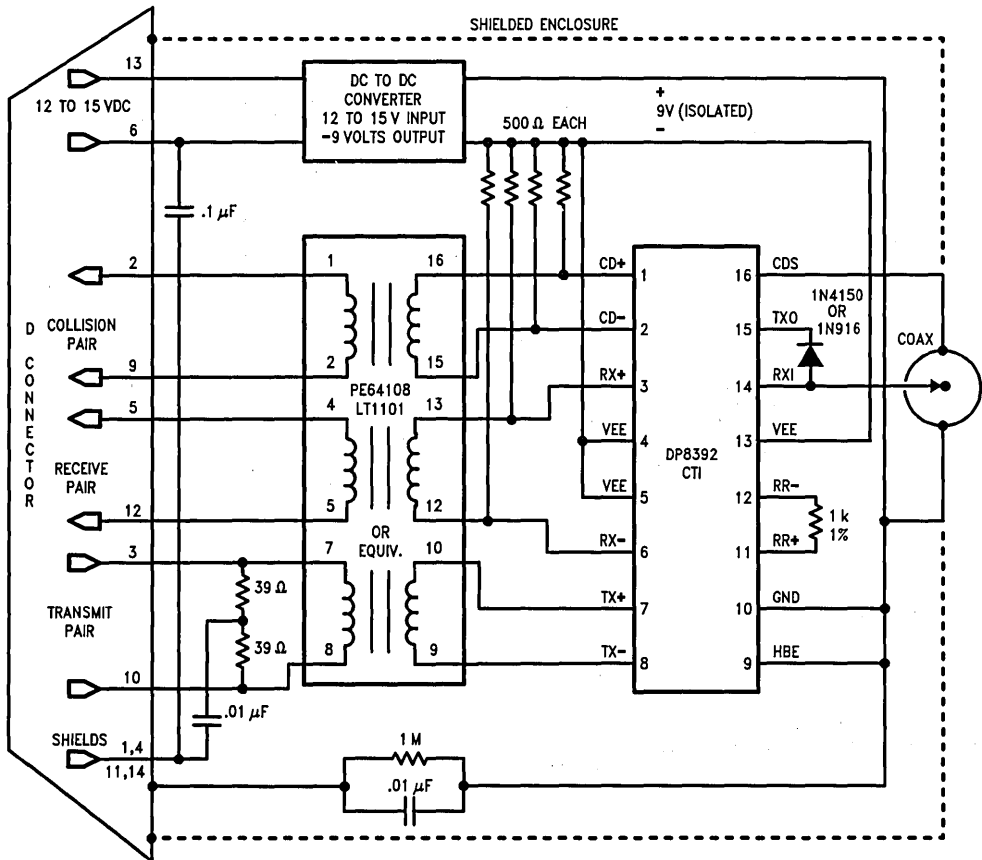


FIGURE 8. An Ethernet MAU Implementation with the CTI

TL/F/8689-12

### CHEAPERNET APPLICATION WITH THE DP8391 AND DP8392

The pin assignment of both the CTI and the SNI are designed to minimize the crossover of any printed circuit traces. Some of the components needed for an Ethernet like interface are not needed for Cheapernet. For instance, Cheapernet's relaxed load capacitance (8 pF, compared with 4 pF for Ethernet) obviates the need for an external capacitance isolation diode at TXO. Also, since the transceiver drop cable is not used in Cheapernet, there's no need for the 78Ω termination resistors. Moreover, without the 78Ω loading on the differential outputs, the pull-down resistors for both the CTI's collision and receive drivers and the SNI's transmit driver can be larger to save power. These resistors can be 1.5k instead of 500Ω for the CTI and 500Ω instead of 270Ω for the SNI.

The 20 MHz crystal connection to the SNI requires special care. The IEEE 802.3 standard requires a 0.01% absolute accuracy on the transmitted signal frequency. An external capacitor between the X1 and X2 pins is normally needed to get the required frequency range. Section 3.1 of the data sheet describes how to choose the value of this capacitor.

The SNI also provides loopback capability for fault diagnosis. In this mode, the Manchester encoded data is internally diverted to the decoder input and sent back to the controller. Thus both the encoding and the decoding circuits are tested. The transmit differential output driver and the differential input receiver circuits are disabled during loopback. This mode can be enabled by a TTL active high input at pin 7.

Two different modes, half step and full step, can be selected at the SNI's transmit output. The standards require half step mode of operation, where the output goes to differential zero during idle to eliminate large idle currents through the pulse transformers. On the other hand, the differential output remains in a fixed state during idle in full step mode. The SNI thus can be used with transceivers which work in either mode. The two different modes can be selected with a TTL input at pin 5.

Figure 9 shows a typical Cheapernet connection diagram using the DP8391 and the DP8392.

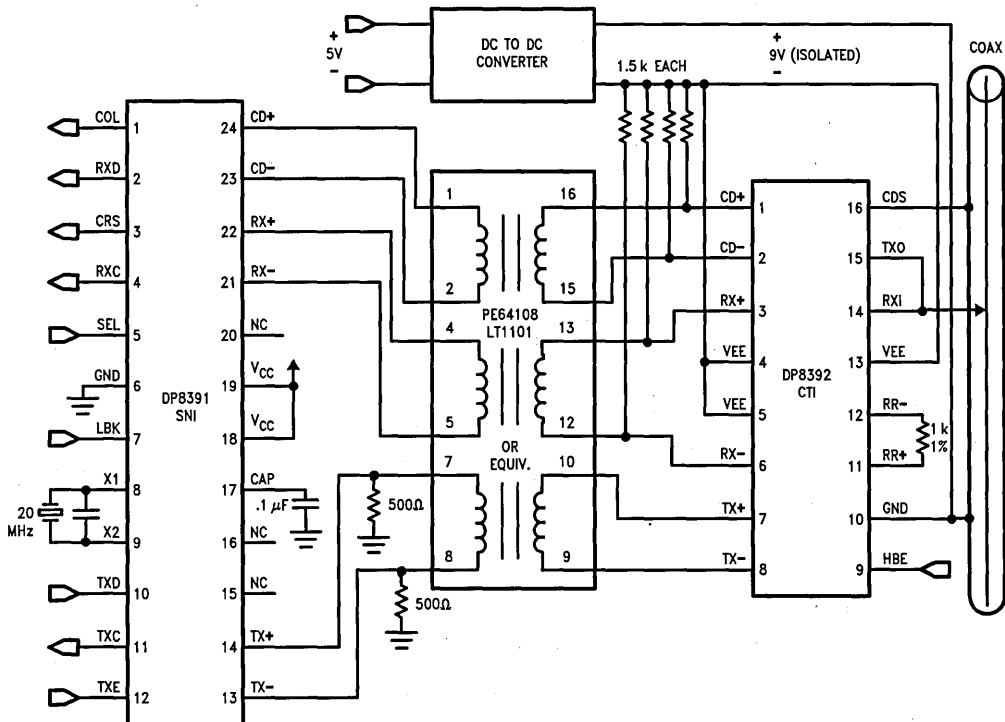


FIGURE 9. Cheapernet Connection Diagram

TL/F/8689-13

The power isolation is similar here as in the Ethernet application, except the DC input is now usually 5V instead of 12V. Hybrid DC to DC converters are also being developed

for this application (Ex: Pulse Engineering PE64381). Figure 10 shows a discrete implementation with 5V input and -9V output.

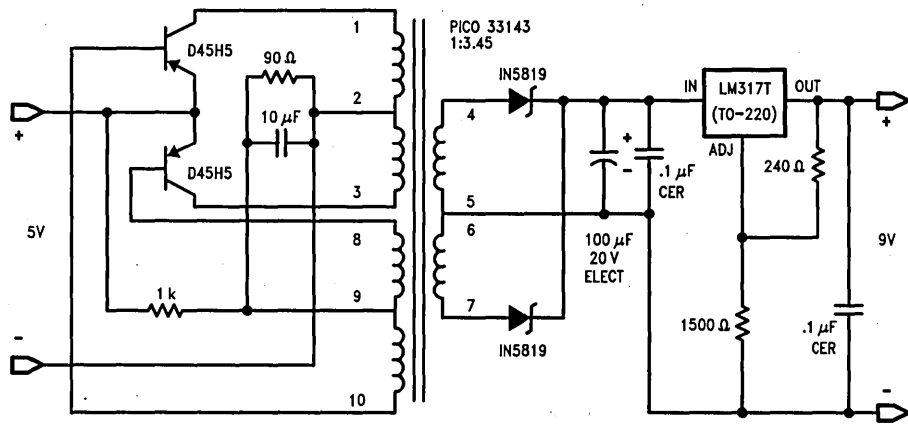


FIGURE 10. DC to DC Converter (5V to -9V)

TL/F/8689-14

# Interfacing the DP8392 to 93Ω and 75Ω Cable

National Semiconductor  
Application Note 620  
Mohammed Rajabzadeh



The DP8392 Ethernet Coaxial Transceiver Interface (CTI) is designed primarily for 10BASE2 and 10BASE5 applications which use 50Ω coaxial cable. However, with minor modifications it is possible to use this transceiver with larger impedance cables. This article shows how to use the DP8392 with 75Ω or 93Ω cable. The trade off is that segment span is reduced to accommodate for higher series DC resistance of these cables. The CTI is a current driver. The two important factors that must be handled properly in using the chip with 75Ω and 93Ω cables are the dynamic range of the transmitter and collision detection levels.

## DYNAMIC RANGE

The dynamic range of the transmitter is important in the following case:

Suppose two stations collide with one-another. To detect collisions properly, each station must sink at least as much DC current as it would in a non-collision case. This would mean that with the 93Ω cable when a collision occurs the chips should be able to sustain approximately -4V DC level. If the signals from the colliding stations are in phase the AC signal could be 8V peak to peak.

The DP8392's transmitter clamps before it pulls to -8V. However, when it clamps it also changes the duty cycle enough to sustain the -4V DC collision level.

An internal diode is included in series with the transmitter's output to isolate its capacitance and thereby minimizing the tap capacitance. For more dynamic range margin, it is recommended that external isolating diodes at the transmitter output not be used. It is also advisable to design the power supply to operate at the higher end of the 8.55V to 9.45V range.

## COLLISION LEVELS—RECEIVE MODE

In order to understand the concerns with collision levels, it is necessary to calculate the levels for Cheapernet (10BASE2) 50Ω cable (RG58AU) as an example.

### 50Ω Cable Example (RG58A/U)

Table I shows the parameter values that are used in calculating the collision levels. Please note that all the levels in this article are for receive mode collision detection.

TABLE I. Assumptions and Definitions

|           |                                                                       |                        |            |
|-----------|-----------------------------------------------------------------------|------------------------|------------|
| $R_T$     | = Termination Resistor at 20°C                                        | = 50 ± 1%              | 802.3      |
| $t_T$     | = Temp. Coef. of the Terminator                                       | = 0.0001/°C            | ASSUMPTION |
| L         | = Maximum Segment Length                                              | = 185m                 | 802.3      |
| $R_{DC}$  | = Maximum Cable DC Res. at 20°C                                       | = 0.0489Ω/m            | BELDEN     |
| $t_c$     | = Temp. Coef. of Copper                                               | = 0.004/°C             | PHYSICS    |
| $T_m$     | = Maximum Cable Temp.                                                 | = 50°C                 | ASSUMPTION |
| SR        | = Step Response at Max Cable Length                                   | = 0.98                 | NATIONAL   |
| $R_C$     | = Max Connector Res./Station                                          | = 0.0034Ω              | MIL SPEC   |
| $I_{B+}$  | = Max Positive Bias Current                                           | = 2 μA                 | 802.3      |
| $I_{B-}$  | = Max Negative Bias Current                                           | = 25 μA                | 802.3      |
| $I_{max}$ | = Max DC Drive Current                                                | = 45 mA                | 802.3      |
| $I_{min}$ | = Min. DC Drive Current                                               | = 37 mA                | 802.3      |
| $R_O$     | = Non Transmitting Output Impedance                                   | = 100 kΩ               | 802.3      |
| N         | = Max Nodes per Segment                                               | = 30                   | 802.3      |
| SK        | = Skew Factor, Effect of Encoder Skew on DC Level<br>= (SKEW × 4)/100 | = 0.02 for 0.5 ns Skew | 802.3      |
| $R_S$     | = Max DC Loop Res. of a Segment                                       |                        | DEFINITION |
| $R_L$     | = Load Resistance Seen by a Driver                                    |                        | DEFINITION |
| SEO       | = Sending End Overshoot                                               | = 0.08                 | ASSUMPTION |

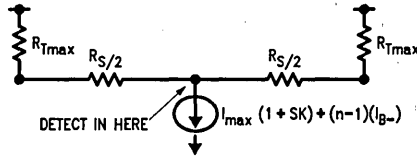


The collision levels that need to be calculated are  $V_{max}$  and  $V_{min}$ . The  $V_{max}$  or "no detect" level is the maximum DC voltage generated by one node. The worst case here occurs when the transmitting node is at the center of a maximum length cable, and the collision is being detected either by itself or by a station right next to it. On the other hand, the  $V_{min}$  or "must detect" level is the minimum DC voltage generated by two minimum stations transmitting at one end of a

maximum length cable, and the collision is being detected by a node on the other side of the cable.

The filter impulse response is not included in these calculations since it is mutually exclusive with the Sending End Overshoot. If the impulse response is larger than the Sending End Overshoot, the exceeding portion should be added on to the limits.

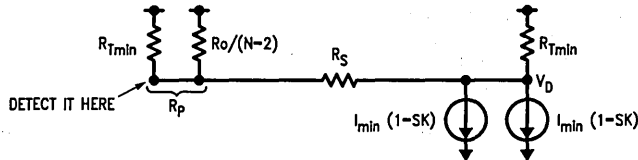
**Maximum Non Collision Level  $V_{max}$  (No-Detect)—Receive Mode—50Ω Cable**



TL/F/10444-1

|            |                                                                             |                     |
|------------|-----------------------------------------------------------------------------|---------------------|
| $R_{Tmax}$ | $= R_T \times 1.01 \times [(Tm - 20) \times t_T + 1]$                       |                     |
| $R_S$      | $= R_{DC} \times L \times [(Tm - 20) \times t_c + 1] + N \times R_C$        |                     |
| $R_L$      | $= (R_{Tmax} + R_S/2)/2$                                                    |                     |
| $V_{max}$  | $= [I_{max} \times (1 + SK) + (N - 1)(I_{b-})] \times R_L \times (1 + SEO)$ |                     |
| $R_{Tmax}$ | $= 50 \times 1.01 \times [(50 - 20) \times 0.0001 + 1]$                     | $= 50.652\Omega$    |
| $R_S$      | $= 0.0489 \times 185[(50 - 20) \times 0.004 + 1] + 30 \times 0.0034$        | $= 10.234\Omega$    |
| $R_L$      | $= (50.652 + 10.234/2)/2$                                                   | $= 27.885\Omega$    |
| $V_{max}$  | $= [45 \times 1.02 + 29 \times 0.025] \times 27.885 \times 1.08$            | $= 1404 \text{ mV}$ |

**Minimum Collision Level  $V_{min}$  (Must-Detect)—Receive Mode—50Ω Cable**



TL/F/10444-2

|            |                                                                    |                     |
|------------|--------------------------------------------------------------------|---------------------|
| $R_p$      | $= \text{NEAR END SHUNT RESISTANCE}$                               |                     |
|            | $= [R_O/(N - 2)]/R_{Tmin}$                                         |                     |
| $R_{Tmin}$ | $= R_T \times 0.99$                                                |                     |
| $V_D$      | $= \text{TRANSMITTER'S END DC VOLTAGE}$                            |                     |
|            | $= 2 \times I_{min} \times (1 - SK) \times [R_{Tmin}/(R_S + R_p)]$ |                     |
| $V_{min}$  | $= V_D \times [R_p/(R_S + R_p)] \times SR$                         |                     |
| $R_p$      | $= [100k/28]/[(50 \times 0.99) = 3571/49.5$                        | $= 48.823\Omega$    |
| $V_D$      | $= 2 \times 37 \times 0.98 \times [49.5/(10.234 + 48.823)]$        | $= 1952 \text{ mV}$ |
| $V_{min}$  | $= 1952 \times [48.823/(10.234 + 48.823)] \times 0.98$             | $= 1581 \text{ mV}$ |

The calculations show that the  $V_{max}$  and  $V_{min}$  are properly placed outside the collision threshold range of the DP8392 (1450 mV to 1580 mV).

### 93Ω Cable Collision Level Calculation

A few parameters need to be changed when using a different impedance cable. Here are those parameters for 93Ω cable (RG62A/U TYPE, BELDEN 9269);

TABLE II

|          |                                 |              |        |
|----------|---------------------------------|--------------|--------|
| $R_T$    | = termination resistor at 20°C  | = 93 ± 1%    |        |
| L        | = maximum segment length        | = 130m       |        |
| $R_{DC}$ | = maximum cable DC res. at 20°C | = 0.1437 Ω/m | BELDEN |

Considering the new values the  $V_{max}$  and  $V_{min}$  levels are;

#### Maximum Non Collision Level $V_{max}$ (No Detect)—Receive Mode—93Ω Cable

|            |                                                                      |               |
|------------|----------------------------------------------------------------------|---------------|
| $R_{Tmax}$ | = $93 \times 1.01 \times [(50 - 20) \times 0.0001 + 1]$              | = 94.212Ω     |
| $R_S$      | = $0.1437 \times 130[(50 - 20) \times 0.004 + 1] + 30 \times 0.0034$ | = 21.025Ω     |
| $R_L$      | = $(94.212 + 21.025/2)/2$                                            | = 52.362Ω     |
| $V_{max}$  | = $[45 \times 1.02 + 29 \times 0.025] \times 52.362 \times 1.08$     | = 2636.692 mV |

#### Minimum Collision Level $V_{min}$ (Must Detect)—Receive Mode—93Ω Cable

|           |                                                               |               |
|-----------|---------------------------------------------------------------|---------------|
| $R_P$     | = $[100k/28]/(93 \times 0.99) = 3571/92.07$                   | = 89.756Ω     |
| $V_D$     | = $2 \times 37 \times 0.98 \times [92.070/(21.025 + 89.756)]$ | = 3646.396 mV |
| $V_{min}$ | = $3646.396 \times [89.756/(21.025 + 89.756)] \times 0.98$    | = 2895.272 mV |

### 93Ω IMPLEMENTATION WITH DP8392

Figure 1 shows the connection diagram with 93Ω cable (100 meters and 30 stations). The design parameters defined below are summarized in Table III. The resistor divider ratio needs to be calculated to attenuate the receiver input signal. The two resistors  $R_1$  and  $R_2$  should center the calculated thresholds (2636 mV to 2895 mV) to the internal level of DP8392 (1450 mV to 1580 mV).

The resistor divider and the capacitor  $C_p$ , Figure 1, ( $C_p$  includes the RXI input capacitance, typically 1 pF, and the pc trace capacitance associated with it) form a low pass filter effect. It may be necessary to add the capacitor  $C_c$  (capacitor  $C_c$  creates a high pass effect) to compensate the low pass effect. The equation to calculate the capacitor  $C_c$  is;

$$C_c \times R_2 = C_p \times R_1$$

It is also necessary to add the resistor  $R_3$  ( $R_3 = R_1 // R_2$ ) in series with the CDS pin. This will assure that the voltage drop due to the biasing currents into CDS and RXI pins are duplicated.

To check the design;

$$[54.8k/(54.8k + 45.2k)] \times 2636 \text{ mV} = 1444 \text{ mV}$$

$$[54.8k/(54.8k + 45.2k)] \times 2895 \text{ mV} = 1586 \text{ mV}$$

The DP8392's internal collision range is within this window.

### 75Ω CABLE IMPLEMENTATION

This method can also be successfully implemented for 80 meters of 75Ω cable (RG59/U BELDEN 8241). The collision thresholds are 2127.8 mV and 2339.6 mV. The corresponding  $R_1$  and  $R_2$  values are 67.8 kΩ and 32.2 kΩ respectively. Table IV summarizes the design parameters.

| CABLE    | BELDEN RG62A/U Type 93Ω Cable |
|----------|-------------------------------|
| L        | 130 meters                    |
| $R_{DC}$ | 0.1437 Ω/m                    |
| N        | 30                            |
| $R_1$    | 54.8k                         |
| $R_2$    | 45.2k                         |

| CABLE    | BELDEN RG59/U 75Ω Cable |
|----------|-------------------------|
| L        | 80 meters               |
| $R_{DC}$ | 0.1894 Ω/m              |
| N        | 30                      |
| $R_1$    | 67.8k                   |
| $R_2$    | 32.2k                   |

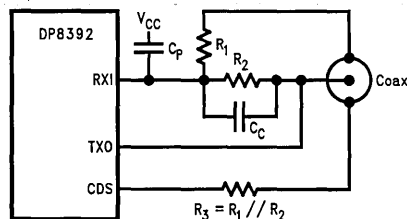


FIGURE 1

TL/F/10444-3

# Low Power Ethernet with the CMOS DP83910 Serial Network Interface

National Semiconductor  
Application Note 622  
William Harmon



AN-622

## INTRODUCTION

This application note discusses the features of, and implementation techniques for, National Semiconductor's CMOS Serial Network Interface (SNI), the DP83910. Also, a comparison of the CMOS SNI to National's bipolar SNI (DP8391) on several key issues will be provided. In general, the DP83910 provides a low power Attachment Unit Interface (AUI) for a Carrier-Sense Multiple Access with Collision Detect (CSMA/CD) Ethernet system. In fact, when used in conjunction with National Semiconductor's Network Interface Controller (NIC, DP8390) and Coaxial Transceiver Interface (CTI, DP8392), the DP83910 provides for a complete IEEE 802.3 Ethernet and/or thin wire Ethernet solution, as shown in *Figure 1*.

## FUNCTIONAL DESCRIPTION OF THE DP83910

The CMOS SNI operates as an interface between an Ethernet transceiver and a local area network data controller. A functional block diagram of the DP83910 is shown in *Figure 2*. The primary function of this interface is to perform the encoding and decoding that is necessary for the differential pair Manchester encoded data of the transceiver and the Non-Return-to-Zero (NRZ) serial data of the NIC to be compatible with each other. In the case of a transmission, the SNI translates the NRZ serial data from a network controller's transmit data line into differential pair Manchester encoded data on a transceiver's transmit pair. In order to

perform this operation, the NRZ bit stream is first received by the Manchester encoder block of the SNI. Once the bit stream is encoded, it is transmitted out differentially on to the transmit differential pair through the transmit driver. When a reception takes place, the differential receive data from a transceiver is converted from Manchester encoded data into NRZ serial data and a receive clock, which are passed to the receive data and receive clock inputs of the Network Interface Controller. In executing this sequence, the DP83910's data receiver takes the Manchester data from the differential receive lines and passes it to the phase locked loop (PLL) decoder block. The PLL block then decodes the data and generates a data receive clock and a stream of NRZ serial data, which is presented to the NIC. In the case of National Semiconductor's Network Interface Controller, the DP8390, the serial NRZ signals are called TXD and RXD.

In addition to performing the Manchester encoding and decoding function, the DP83910 also provides several important network signals to the network controller. A diagram of the interface between National Semiconductor's NIC and the CMOS SNI can be found in *Figure 3*. The first of these signals is carrier sense (CRS), which indicates to the controller that data is present on the SNI's receive differential pair. Secondly, the SNI provides the network controller with a collision detection signal (COL), which informs the controller that a collision is taking place somewhere on the net-

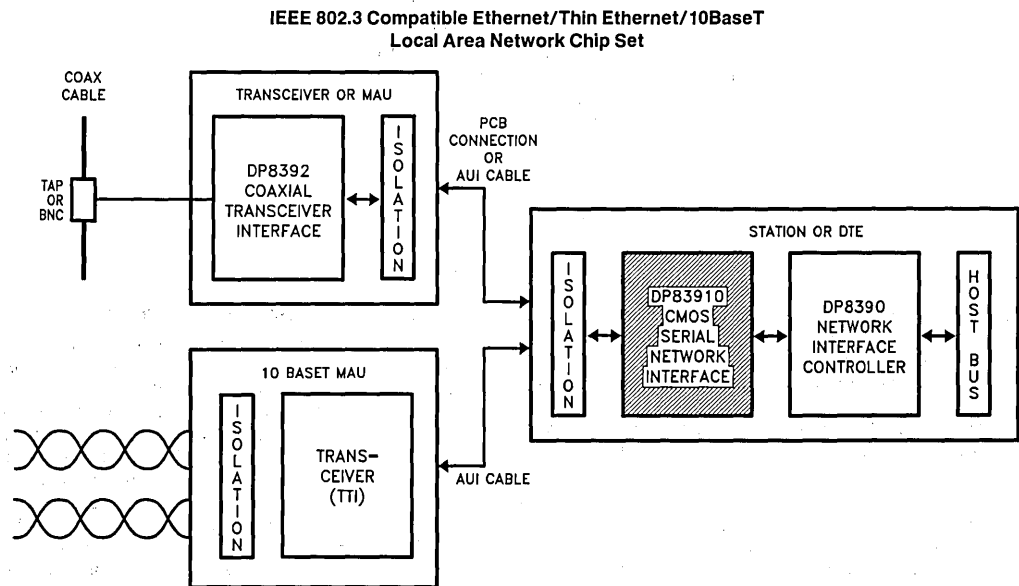


FIGURE 1. A Block Level Diagram of an Ethernet Node

TL/F/10446-1

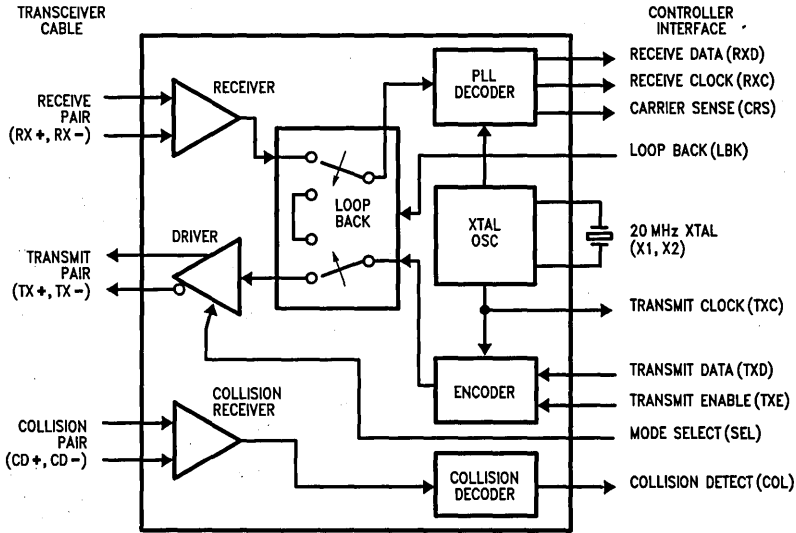


FIGURE 2. DP83910 Block Diagram

TL/F/10446-2

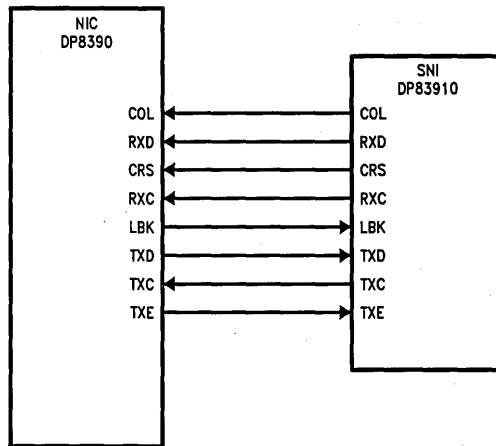


FIGURE 3. Interface between the DP8390 and DP83910

TL/F/10446-3

work. The SNI itself is informed of the collision when its collision receiver detects a 10 MHz signal on the differential collision input pair. Finally, the DP83910 provides both the receive and transmit clocks (RXC and TXC, respectively). The transmit clock is a divide by two derivative of the SNI's oscillator inputs (X1 and X2), while the receive clock is generated directly from the frequency of the input data to the PLL.

The DP83910 can also be placed in a loopback mode, in order to check the SNI's receive and transmit interface to the network controller. In loopback, as pictured above, the SNI's Manchester encoder block is essentially connected directly to the PLL decoder block. This allows for the validation of the Manchester encoding and decoding process without the variable of random network traffic. The SNI is placed in loopback mode when the loopback pin (LBK) is driven high.

**COMPARING THE DP83910 WITH THE DP8391**

The DP83910 is basically a CMOS version of the existing National Semiconductor bipolar SNI, the DP8391. The functionality of the two parts is identical. However, there are a few differences that exist between the two parts, in spite of the fact that they can be implemented as pin for pin compatible. The most fundamental difference between the two parts is the process under which each is manufactured. The DP83910 SNI is fabricated in a CMOS process, while the DP8391 is made in a bipolar process. As a result of this, the level of average power supply current needed by the DP83910 is approximately 75 percent less than the 270 mA required by the DP8391. Another significant difference between the two parts is the CMOS SNI's need for a pulse transformer to be placed between all of its differential signals and those of the transceiver, regardless of whether a drop cable or thin wire Ethernet configuration is being imple-

mented. This is necessary due to the fact that the CMOS process will not guarantee the IEEE 802.3 16V fail safe specification if no isolation is provided to the differential signals that go to the AUI cable. One consequence of the transformer requirement is that National Semiconductor defines the AUI interface at the transceiver side of the transformer and only guarantees the correct operation of the CMOS SNI when the pulse transformer is employed in the system.

In addition to the above process related differences, there are still two non-process related differences, which need to be mentioned. First, the phase locked loop in the bipolar SNI is digital, while the phase locked loop of the CMOS SNI is analog. This is functionally transparent when designing with the DP83910; however, it does provide for a significant savings in power consumption. Finally, it should be noted that pin 17 (TEST) on the bipolar SNI is required to be tied to ground through a capacitor, while the same pin on the CMOS SNI can either be implemented in the same manner or connected directly to ground. A list of all the above mentioned differences can be found in Table I.

#### DESIGNING WITH THE DP83910

In developing the DP83910, National Semiconductor performed extensive testing in its own Local Area Network Laboratory to assure that the CMOS SNI would provide an easily implemented low power controller/transceiver interface for Ethernet system designers. This development and testing assured that the DP83910 was IEEE 802.3 and Ethernet compatible, able to interface with industry standard transceivers (Ethernet, Twisted Pair Ethernet, and Fiber Optic Ethernet), and is capable of having the National Semiconductor DP8391 as a pin-for-pin replacement. In *Figures 4* and *5*, two methods of implementing the DP83910 with the DP8392 are demonstrated. One significant feature of both designs is that it is possible to directly substitute a DP8391 for the CMOS SNI and maintain the same functional quality.

#### The DP83910 Transmitter Operation

When operating as a transmitter, the DP83910 combines NRZ data received from the controller with a clock signal, which the SNI generates, and encodes them into a Manchester serial bit stream. This encoded signal then appears differentially at the SNI's TX± output. In Ethernet (10Base5) applications, this signal is sent to the transceiver or the Medium Attachment Unit (MAU) through an AUI transceiver cable. This cable, which can be up to 50 meters

in length, typically consists of four individually shielded twisted wire pairs (TX±, RX±, CD±, and PWR/GND), which are covered by an additional overall shield. The transmit signal pair, which has a differential characteristic impedance of 78Ω, should be terminated at the receiving end of the cable. It should be noted that each of the TX+ and TX- source follower outputs needs to be connected to ground through a 270Ω pull down resistor.

When employing the CMOS SNI, it is important to place a pulse transformer between the differential transmit pair on the DP83910 and the differential transmit signal on the AUI cable or CTI, as shown in *Figures 4* and *5*. This transformer is required in order to provide the necessary isolation for the CMOS SNI to meet the IEEE 802.3 16V fail safe specification. However, the pulse transformer does reduce the transmission of noise onto the transceiver cable. Also, it should be noted that more inductive transformers will decrease the magnitude of the undershoot. Furthermore, it is imperative that the designer guarantee the inductive load seen between the DP83910's AUI interface and the CTI receiver be greater than 27 μH. Transformers with 50 μH to 150 μH loading, such as the Pulse Engineering PE64103 and Nano Pulse NP5417, are recommended, since they will minimize the inductive undershoot on the SNI's TX± output pair and reduce the noise seen by the CTI's differential transmit input pair. It is important that the selected pulse transformer doesn't excessively increase the rise and fall time nor lower the output amplitude despite the fact that it reduces the undershoot.

The DP83910 provides both half and full step modes. The IEEE 802.3 standard requires the use of half step mode, in which the transmit output goes to differential zero in idle. In full step mode, the transmitter enters idle and stays at a fixed level. This will eventually allow the pulse transformer to completely saturate. The desired mode of operation is chosen through the Mode Select pin (SEL) on the SNI.

#### The DP83910 Data Receiver Operation

While performing reception, the CMOS SNI receives differential Manchester encoded serial data and converts it into NRZ serial data and a receive clock. The Manchester encoded data, which is received from the CTI or AUI cable, must be isolated before it reaches the SNI. Hence, the DP83910 requires that there be a pulse transformer on the SNI's side of the AUI interface. The actual employment of this transformer can be seen in both *Figures 4* and *5*. This

TABLE I. Comparison of the DP8391 and DP83910

|                                                     | DP8391                           | DP83910                        |
|-----------------------------------------------------|----------------------------------|--------------------------------|
| Process                                             | Bipolar                          | CMOS                           |
| Power Consumption<br>(Typical)                      | 270 mA                           | 70 mA                          |
| Pulse Transformer<br>(At DTE Side of AUI Interface) | Optional                         | Required                       |
| Phase Locked Loop                                   | Digital                          | Analog                         |
| Pin 17                                              | PLL Filter/Capacitor<br>Required | Test Pin/Capacitor<br>Optional |

2-50

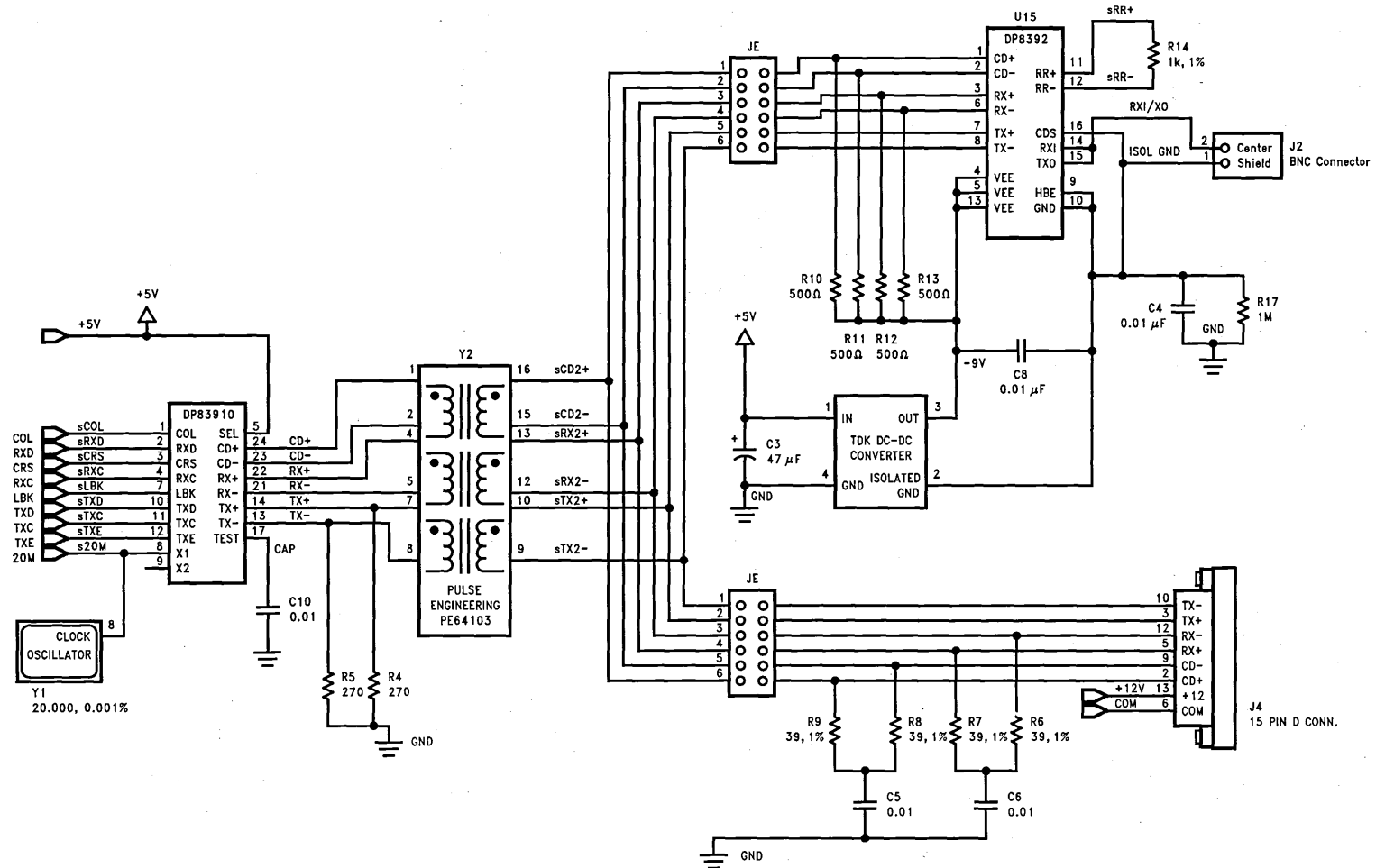


FIGURE 4. Interface for Ethernet and Thin Wire Ethernet

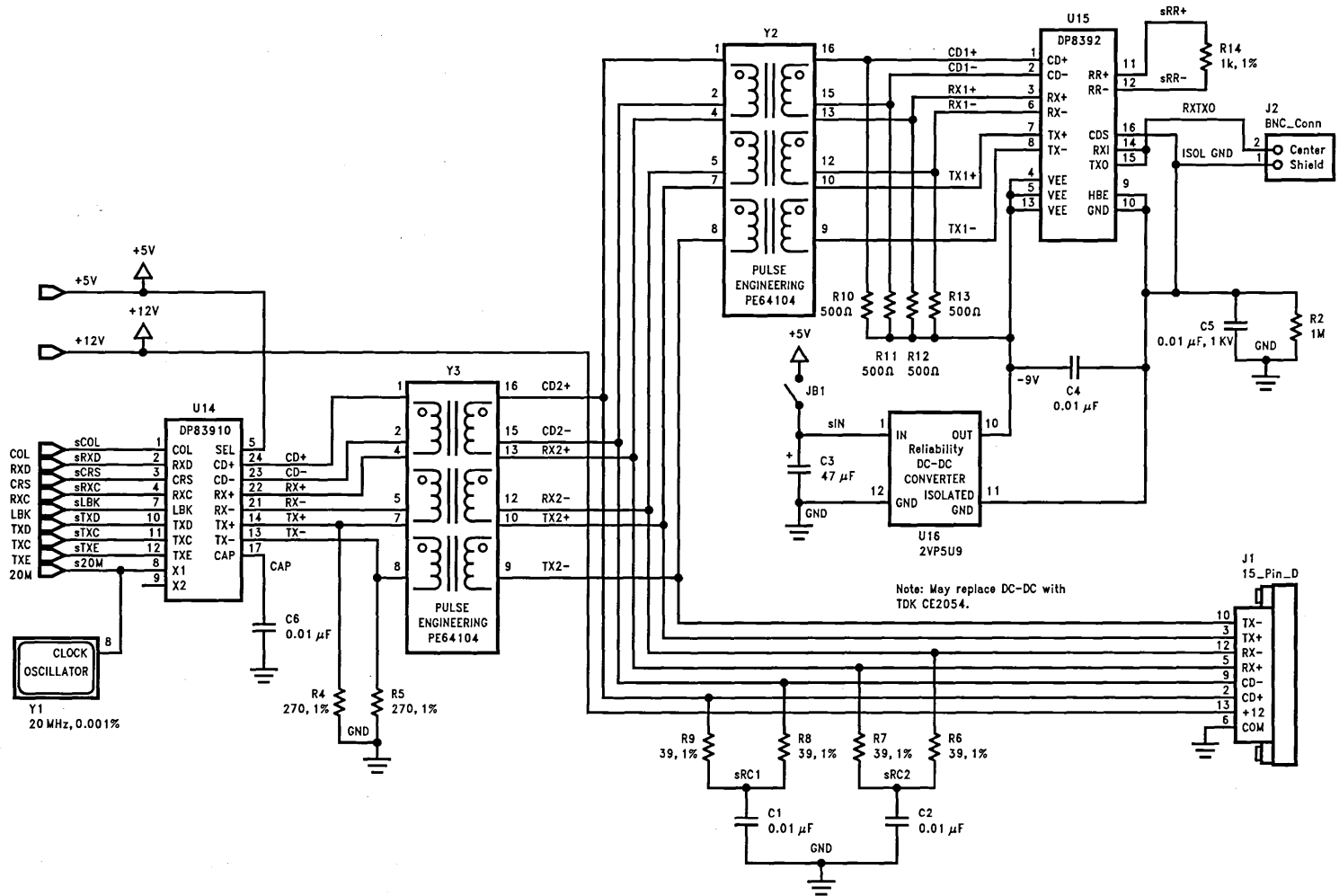


FIGURE 5. Single Switch Solution for Ethernet and Thin Wire Ethernet

TL/F/10446-6

transformer is mandatory and it forms part of the internal DC biasing circuit used for the differential receivers. Furthermore, the transformer is also needed to isolate the transceiver cable against the 16V voltage fault specification in the IEEE 802.3 standard. The performance of the differential receiver is not greatly affected by the selection of a pulse transformer. As a result, the pulse transformer selected for the transmitter design will also work correctly for the RX± data receiver. It should be noted here that the collision receiver is very similar to the data receiver and requires the same isolation. The collision input will be discussed more in the following section.

Once the data arrives at the receiver inputs of the SNI, it is amplified and then decoded by the analog phase locked loop, which can receive Manchester data with ±20 ns of random jitter. During the decoding process, the incoming signal is converted into NRZ data and a receive clock, which are sent to a network controller. Also, the differential data receiver has a built in filter to provide a static noise margin. This filter enables the SNI to reject signals that do not exceed the input squelch voltage and have less than a 30 ns pulse width.

Furthermore, since the DP83910 and pulse transformer constitute the AUI interface, the physical connection between the AUI and the MAU interfaces is defined as being on the MAU side of the pulse transformer. In light of this, it is permissible, when incorporating the CMOS SNI in a thin wire Ethernet application, to have a 78Ω resistance appear across the differential receive and collision inputs to the CTI, as shown in *Figure 5*.

#### The DP83910 Collision Pair Operation

In addition to the data receiver, the DP83910 also provides a differential receiver for the collision pair, which is driven by the transceiver. This 10 MHz active signal, from the AUI Interface, is converted to a TTL signal, digitally stretched, and sent to the controller as the Collision Detect Output (COL). Just as with the data receiver, the differential collision receiver has a built in filter that rejects pulses that do not exceed the input squelch voltage level and have a pulse width less than 30 ns.

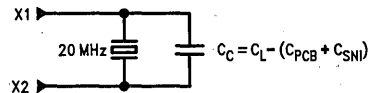
#### Optimal Ethernet and Thin Wire Ethernet Interface

If it is necessary to design a LAN board that minimizes the number of switching devices (jumpers) to alternate between Ethernet and thin wire Ethernet, the solution in *Figure 5* could be employed. This solution, in contrast to the six jumper solution in *Figure 4*, requires only one switch, which enables and disables the power supply to the CTI. In the case of thin wire Ethernet, power would be supplied to the CTI, while during drop cable Ethernet operation the unused CTI would be powered down. Hence, no excessive power is required when thin wire Ethernet is not in use. Furthermore, since there is only one switch, it may be feasible to implement that switch with a transistor as opposed to a jumper. The advantage to using a transistor is that the Ethernet/thin wire Ethernet option can now be made to be software selectable. This is accomplished by developing a control signal, which the software can issue to switch the transistor. Also, in looking at *Figure 5*, it is seen that two pulse transformers are used. The first transformer (Y3) is required by the CMOS SNI, for the reasons previously mentioned. The second pulse transformer (Y2), however, is used to isolate the powered-down CTI from the AUI cable interface, when Ethernet is being used. As in *Figure 4*, the application in *Figure 5* allows the direct substitution of a bipolar SNI, the DP8391, for the CMOS SNI.

#### The DP83910 Oscillator Inputs

The oscillator inputs of the CMOS SNI can be driven with a crystal or an oscillator. In either case, the SNI oscillator must be driven with a 20 MHz signal that provides for the transmitted frequency to be accurate within 0.01% as specified in IEEE 802.3 standard. When using an oscillator, the output of the oscillator should be tied to input X1 of the SNI and the X2 input of the SNI should be left unconnected or grounded. However, the employment of a crystal to generate the 20 MHz signal at the SNI's oscillator inputs requires a great deal of care. The frequency of the crystal is usually measured with a fixed load capacitance ( $C_L$ , typically 20 pF), which is specified in the crystal's data sheet. In order to prevent any distortion in the transmitted frequency, the total capacitance across the crystal's leads should equal its specified load capacitance. The capacitance that is seen by the crystal's leads is the sum of the stray PC board capacitance ( $C_{PCB}$ ) and the capacitance looking into the X1 and X2 inputs ( $C_{SNI}$ ). If this capacitance is smaller than the crystal's load capacitance, a correctional capacitance ( $C_C$ ) can be placed across the crystal's leads. This correctional capacitance would equal the difference between the crystal's load capacitance and the sum of the stray PC board capacitance and the SNI's X1 and X2 input capacitance. It should be noted that the input capacitance of the SNI that is seen across X1 and X2 is approximately a negligible 0.5 pF. *Figure 6* displays a possible crystal setup. The selected crystal should meet the following specifications:

|                    |                     |
|--------------------|---------------------|
| Resonant frequency | 20 MHz              |
| Tolerance          | ±0.001% at 25°C     |
| Stability          | ±0.005% at 0°C–70°C |
| Type               | AT cut              |
| Circuit            | Parallel Resonance  |



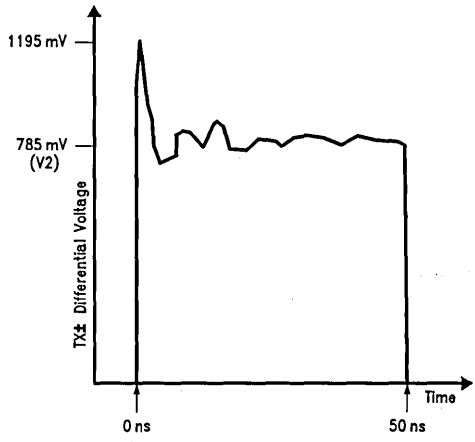
TL/F/10446-5

FIGURE 6. SNI Oscillator Input Circuit

#### Improving Transmitter Overshoot

Upon transitioning from a differential voltage of one polarity to another polarity (i.e., positive to negative), the magnitude of the differential transmit signal will reach a peak value. This peak at the transition points in the differential transmit waveform is referred to as the overshoot voltage. The overshoot voltage of the DP83910 is below the maximum allowable 1315 mV value that appears in the IEEE 802.3 standard. However, the IEEE standard also defines the overshoot voltage to be no greater than 1.12 times the nominal value (IEEE calls this nominal value V<sub>2</sub>). The DP83910 exceeds this particular segment of the overshoot specification, as shown in *Figure 7*. However, exceeding the allowable overshoot voltage value, as the CMOS SNI does, will have no functional affect on a system. Furthermore, the overshoot voltage can be altered to adhere to the IEEE 802.3 specification by placing a capacitor across the differential transmit pair at the primary (SNI side) of the required pulse transformer. This capacitor should be in the range of 40 pF to 50 pF and will not degrade the performance of the CMOS SNI or system in any way. It should also be mentioned that the DP8391, the bipolar SNI, will still be a pin-for-pin replacement for the CMOS SNI, in a design which employs the capacitor for improving the overshoot.





TL/F/10446-7

FIGURE 7. TX± Differential Overshoot Voltage

# Measuring Ethernet Tap Capacitance

National Semiconductor  
Application Note 757  
Larry Wakeman



## INTRODUCTION

When a node is added to an Ethernet network, its nodal capacitance changes the impedance of the cable at the point of connection to the cable. The impedance change causes a reflection of the Ethernet waveform, which distorts the waveform. The more the capacitance the greater the distortion, and eventually with large enough node capacitances the Ethernet signal could become so distorted that the packet data would become corrupted when decoded by a network node. For this reason the IEEE802.3 standard specifies a maximum value of capacitance that a node may add to the network, as well as a minimum node to node distance spacing. Since the capacitance of a node includes stray inductances, the effective capacitance of a node connection cannot be measured simply by using a capacitance meter. This note presents the method for measuring capacitance of an Ethernet tap for 10BASE5 or a BNC "T" for 10BASE2.

## THE STANDARD'S REQUIREMENTS

To properly make the measurement, it is important to understand how the standard specifies the capacitance of a node. To quote the IEEE802.3 standard:

**8.3.1.1 Input Impedance:** The shunt capacitance presented to the coaxial cable by the MAU circuitry (not including the means of attachment to the coaxial cable) is recommended to be no greater than 2 pF. The resistance to the coaxial cable shall be greater than 100 k $\Omega$ .

The total capacitive load due to MAU circuitry and the mechanical connector as specified in 8.5.3.2 shall be no greater than 4 pF.

These conditions shall be met in the power-off and power-on, not transmitting states (over the frequencies BR/2 to BR).

The magnitude of the reflection from a MAU shall not be more than that produced by a 4 pF capacitance when measured by both a 25 ns rise time and 25 ns fall time waveform. This shall be met in both the power-on and power-off, not transmitting states.

To summarize the maximum allowable capacitance specifications for both Thinwire and Thickwire Ethernet the following table is provided.

TABLE I. Maximum Capacitance Allowed in IEEE802.3

| Standard | Electrical Circuitry | Mechanical Connector |
|----------|----------------------|----------------------|
| 10BASE5  | 2 pF                 | 2 pF                 |
| 10BASE2  | 4 pF                 | 4 pF                 |

Note: Thickwire or Thick Ethernet refers to 10BASE5 and Thinwire or Thin Ethernet refers to 10BASE2.

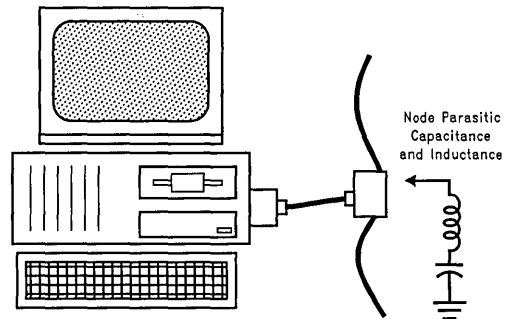


FIGURE 1. Simple Model of the Parasitics Presented to the Ethernet Cable

## THE TEST METHOD

Due to the nature of the capacitance of a DTE (Data Terminal Equipment), rather than perform a simple capacitive measurement using a meter, the capacitance of the network node is more accurately measured by testing it in an environment where the actual signal reflection caused by the capacitance of a node attachment is measured when applying a typical Ethernet signal. The magnitude of the reflection is then correlated to an equivalent capacitance. This is the most appropriate method, since it is the signal degradation due to the capacitive load that is the important consideration in defining the above specifications.

With the above in mind, the test is performed by first measuring the reflection caused by the attachment of a node. Then the DTE is replaced with a reference variable capacitor, and the capacitor's value is adjusted until the capacitance that causes the same size reflection is determined. The capacitance of the node is therefore the same as the reference capacitance value that causes the same amplitude reflection.

**TEST SETUP AND CABLE**

An example test configuration which measures the capacitance of the Thickwire Ethernet is shown in *Figure 2*. The waveform applied to the test node is an important consideration in setting up the test, as it will affect the resultant value of capacitance. In particular the rise and fall times must be carefully chosen to reflect the capacitance seen in an Ethernet network, as described in the next section.

The cable lengths and spacing between the scope input and the transceiver's connection are chosen to ensure that the reflection due to the transceiver appears on the flat portion of the test waveform. This allows accurate measurement. The total cable length is equivalent to the full 10BASE5 length of 500m.

An oscilloscope is used to measure the voltage of the reflection. The scope, with a 1 MΩ input impedance, as shown in *Figure 2*, is connected directly to the cable without a probe. This eliminates any errors due to the probe. The distance between transceiver connection point "A" and the scope is set so that the reflections will arrive at the scope right after the signal rise and fall times. Moving point "A" any further makes the reflections smaller in amplitude (cable attenuation) and therefore harder to measure.

On the scope's display measurements are made at the point immediately after the rise time. Reflections are then compared to the ones for known discrete capacitors.

**THE TEST WAVEFORM**

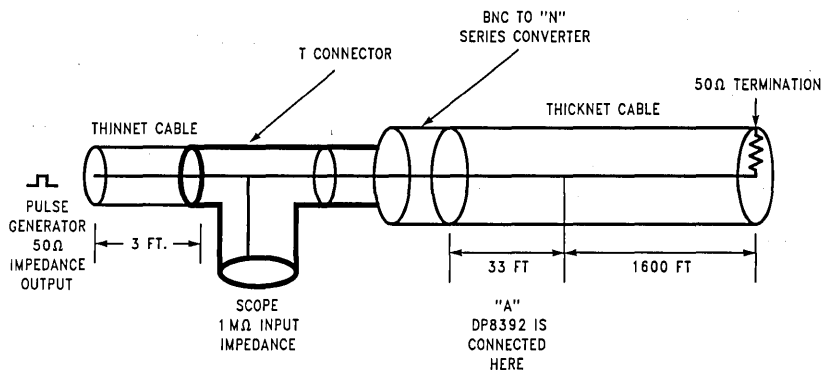
In normal network operation the signal on the coax cable has rise and fall times of 25 ns ± 5 ns (defined by the IEEE802.3 standard). With a purely capacitive load applying signals with faster (or slower) edges cause larger (or smaller) reflections than would be seen on a typical network. If the node were purely capacitive this would not affect the measurement. The larger (or smaller) node reflection for a given parasitic capacitance would track with the reference capacitance's reflection yielding accurate measurements.

However, the node is actually not a pure capacitance, but has some series inductance associated with the network connection as shown in *Figure 1*. The application of signals with faster than 20 ns rise and fall times actually result in an unrealistically low capacitance measurement. This is because the nodes capacitance is buffered by the stray series inductances which reduce the reflection magnitude when compared to the pure capacitance. This correlates to a lower than actual capacitance.

On the other hand applying very slow rise and fall times (slower than 30 ns) result in the measurement of a larger capacitance than actual. This is because the series inductance effects are less than would be seen with a nominal waveform.

Since it is desirable to measure the capacitance in such a way as to correlate to the effective capacitance seen when IEEE802.3 signaling is used, the best compromise choice is to select a 25 ns rise and fall times for this test. (This is the reason for this choice in the actual standard.)

Again, the reason behind this decision is that although the ≥ 30 ns edges indicate larger capacitances a signal with 25 ns edge produces results that more correctly represent the actual effect of the attached node's capacitance.



**FIGURE 2. Test Setup**

TL/F/11163-2

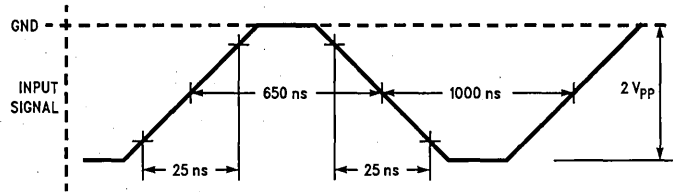
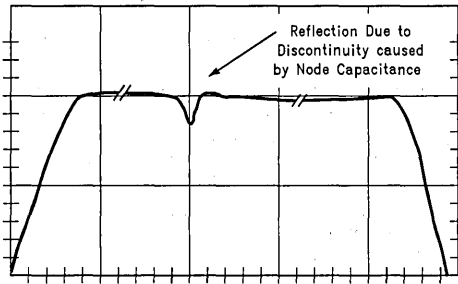


FIGURE 3. Input Test Waveform

TL/F/11163-3

As shown in *Figure 3*, a low frequency trapezoidal signal is used. This will keep the reflections from each edge of the signal well away from the next edge enabling easier measurement. The  $2 V_{pp}$  test input signal is the typical voltage swing on the coax cable in normal operation. In the case of a discrete capacitor the voltage level of the signal may not be important. However, due to the non-linearity of the node and DP8392 capacitance a typical voltage signal should be used following the same rationale as was used for the signal rise and fall times.



TL/F/11163-4

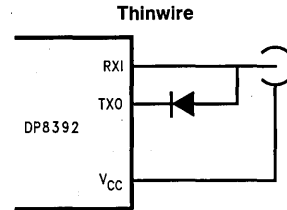
Note: This figure is conceptual. It does not show the waveform details.

FIGURE 4. Example of Reflection

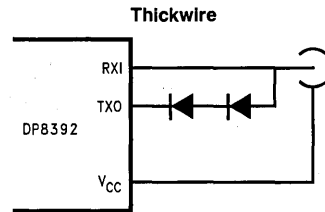
### TEST RESULTS

A special jig was built to connect the ICs to point "A" in *Figure 2*. This greatly improves measurement repeatability. Data repeatability of 0.01 pF is achieved.

Typical data for RXI and TXO capacitances are 1.0 pF and 2.0 pF respectively. Total node capacitance can be reduced to around 1.6 pF with the addition of a small capacitance diode in series with the TXO output, as shown in *Figure 5*. For Ethernet applications two diodes in series can be used instead.



TL/F/11163-5



TL/F/11163-6

FIGURE 5. DP8392 Connection Diagram

### INACCURACIES OF THE CAPACITANCE METER

As stated, in a real network, it is not the node capacitance that creates a problem, but too large a reflection caused by this capacitance. This reflection distorts the cable signal. Therefore the best method of test is to measure the reflection under true network waveforms. By the same analogy capacitance meters which have a test signal frequency that does not correspond to 25 ns rise and fall time do not reveal a true measurement of capacitance, and so capacitive measurements done only with a capacitance meter are usually (almost always) inaccurate to the true effective capacitance as seen by the network cable.

## Ethernet Magnetics Vendors for 10BASE-T, 10BASE2, and 10BASE5

Enclosed is an overview of the magnetics components needed to interface National's Ethernet Products to each of the popular Ethernet cabling schemes.

### 10BASE-T TWISTED PAIR ETHERNET

This discusses the available components to interface National Semiconductor's 10BASE-T Ethernet LAN products to twisted-pair cable. The products offered by NSC include:

- DP83902 ST-NICT™ Serial Twisted Pair Network Interface Controller
- DP83905 AT/LANTIC™ Single Chip Ethernet Controller
- DP83934 SONICT™-T Systems Oriented Network Interface Controller for Twisted Pair
- DP83950 RIC™ Repeater Interface Controller
- DP83955/6 LERIC™ LiE Repeater Interface Controller

The types of solutions from these vendors vary and the designer is encouraged to contact these companies to obtain information on their various solutions. A brief overview of these products is presented here.

The interface from one of National Semiconductor's integrated circuits to the cable consists of the following blocks:

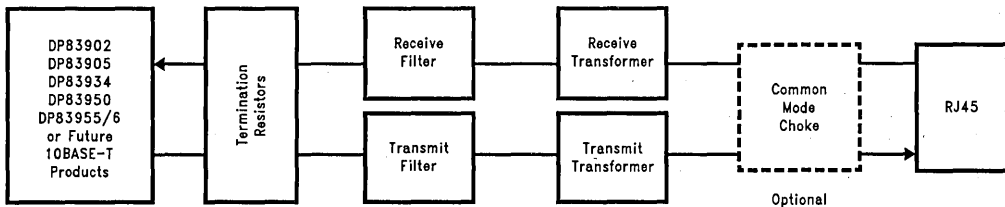
1. Termination resistors used to match the impedance of the twisted-pair interface to the cable.

2. Transmit and Receive Filters which are used to filter out receiver noise, and for the transmitter limit the harmonic content of the output waveform.
3. Transmit and Receive Pulse Transformers. These are required by the 10BASE-T standard to isolate the media from the data terminal equipment (DTE).
4. Optionally a common mode choke which is used to reduce common noise that could be emitted by the 10BASE-T interface. This may be necessary in some applications for meeting FCC or VDE EMI requirements as well as to meet 10BASE-T common mode output voltage noise specifications.

The components offered by the various manufacturers incorporate one or more or all of these. Table I shows some information on the available components. The components listed are primarily those that are more highly integrated. Also mostly DIP version part numbers of these devices are listed, however most vendors have surface mount versions of these products, as well as some products in SIP versions.

Table I is not a complete list of available components. The designer should use this list as a starting point for researching suitable products for his design. The addresses and phone numbers of the vendors listed are shown at the end of this paper in Table IV.

Twisted Pair Cable Interface Blocks



TL/F/11248-1

TABLE I. Partial List of 10BASE-T Transformer-Filter Products

| Part Number              | Filter | Transformer | Choke | Termination Resistor | National* Product  | Package†† Type | Tested** by National |
|--------------------------|--------|-------------|-------|----------------------|--------------------|----------------|----------------------|
| <b>PULSE ENGINEERING</b> |        |             |       |                      |                    |                |                      |
| PE65433                  | ✓      | ✓           | ✓     |                      | 902, 934, 950      | TH             | ✓                    |
| PE65425                  | ✓      | ✓           | ✓     |                      | 902, 905, 934, 950 | TH             |                      |
| PE65434                  | ✓      | ✓           | ✓     |                      | 902, 905, 934, 950 | SIP            |                      |
| PE65438                  | ✓      | ✓           | ✓     | ✓                    | 950, 955, 956      | SIP            | ✓                    |
| PE65483                  | ✓      | ✓           | ✓     | ✓                    | 902, 905, 934      | SMD            |                      |
| PE65443                  | ✓      | ✓           | ✓     | ✓                    |                    | SMD            |                      |
| <b>VALOR</b>             |        |             |       |                      |                    |                |                      |
| PT3877                   | ✓      | ✓           |       |                      | 902, 905, 934, 950 | TH             | ✓                    |
| FL1012                   | ✓      | ✓           | ✓†    |                      | 902, 905, 934, 950 | TH             | ✓                    |
| SF1012                   | ✓      | ✓           | ✓     |                      | 902, 905, 934      | SMD            |                      |
| FL1020/SF1020            | ✓      | ✓           | ✓     | ✓                    | 902, 905, 934, 950 | TH, SMD        | ✓                    |
| FL1085                   | ✓      | ✓           | ✓     | ✓                    | 950, 955, 956      | SIP            | ✓                    |
| FL1059                   | ✓      | ✓           | ✓     | ✓                    | 902, 905, 934      | TH             |                      |
| FL1010-002 (4 Channel)   | ✓      | ✓           | ✓     | ✓                    | 950, 955, 956      | TH             |                      |
| <b>FEE FIL-MAG</b>       |        |             |       |                      |                    |                |                      |
| 78Z1120B/D/F-01          | ✓      | ✓           |       |                      | 902, 905, 934      | TH             |                      |
| 78Z1122B/DF-01           | ✓      | ✓           | ✓     |                      | 902, 905, 934      | TH             |                      |
| 78Z1120B-03              | ✓      | ✓           |       |                      | 902, 905, 934, 950 |                |                      |
| 78Z1122B-11/12/13        | ✓      | ✓           | ✓     | ✓                    | 902, 905, 934, 950 | TH             |                      |
| 78Z1120F-01              | ✓      | ✓           | ✓     |                      | 950, 955, 956      | TH             |                      |
| <b>PCA ELECTRONICS</b>   |        |             |       |                      |                    |                |                      |
| EPA1990                  | ✓      | ✓           |       |                      | 902, 934, 950      | TH             |                      |
| EPA2013D                 | ✓      | ✓           | ✓†    |                      | 902, 934, 950      | TH             |                      |
| EPA2188A                 | ✓      | ✓           | ✓     | ✓                    | 950, 955, 956      | TH             |                      |
| EPA2162                  | ✓      | ✓           | ✓     |                      | 950, 955, 956      | SIP            |                      |

\*902 = DP83902, 950 = DP83950, DP83955, DP83956, 934 = DP83934.

\*\*National has evaluated a sampling of 10BASE-T filter-transformer products for operation with the products listed in the National product column. Other products listed should provide suitable performance but have not been evaluated at this time. These products have been tested to a subset of the 10BASE-T standard when using National Semiconductor's integrated circuits. This testing includes waveshape, amplitude, jitter, and general interoperability. Testing for EMI has not been done as this varies dramatically between test setups and real applications.

†There is a single common mode choke on the transmit channel only.

††TH = Thru-hole; SIP = Single-In-Line Package; SMD = Surface Mount Device

**10BASE2 AND 10BASE5 THIN AND THICK ETHERNET**

The interface for Thin (10BASE2) and Thick (10BASE5) Ethernet to the coaxial cable is nearly the same, and is illustrated by the block diagram in Figure 2. From the AUI (Attachment Unit Interface) to the coax cable there are three major blocks. A major requirement of the interface is the electrical isolation from the cable interface to the AUI. This isolation is required to be 500V for 10BASE2, and 2000V for 10BASE5. Two of the three blocks provide this isolation.

Starting from the AUI, there are 4 pairs of wires. One pair provides power, and the other three are the data and collision signals. The signal pairs connect to a triple pulse trans-

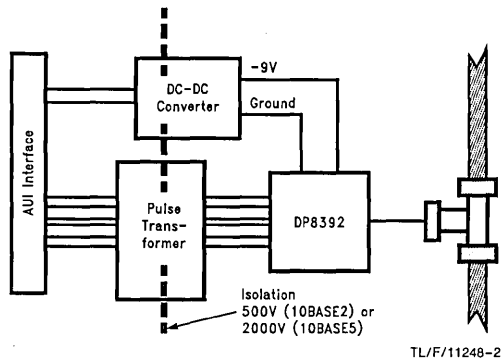
former that provides voltage isolation. These signals then connect to the DP8392, Coax Transceiver Interface (CTI) which converts AUI signaling to coax transmission and reception signals.

The DP8392 is powered from the AUI power pair which is fed from the AUI to a DC to DC Converter. The DC to DC Converter provides the voltage isolation of the power pair required by the IEEE standard, and in 10BASE5 converts the AUI 12V power to -9V required by the DP8392. (For 10BASE2 the DC to DC converter typically converts 5V to -9V).

Besides the DP8392 and a few discrete resistors and capacitors, the major additional magnetic components are the pulse transformer and the DC to DC converter. Both of these components are readily available from a number of sources. Table II shows a selection of manufacturers and their part numbers for Ethernet pulse transformers. All of these components are pin compatible and are available in 16-pin DIP package. Most of these manufacturers also have surface mount versions of these components.

Table III lists two vendors of DC to DC converters. As can be seen there are several different types of converters depending on input voltage and whether an enable function is desired.

Tables II and III are not a complete list of available components. The designer should use these lists as a starting point for researching suitable products for his design. The addresses and phone numbers of the vendors listed are shown at the end of this paper in Table IV.



**FIGURE 2. Coax Ethernet Cable Interface Block Diagram**

**TABLE II. Representative Ethernet Isolation Pulse Transformers for AUI (16-Pin DIP, Triple Transformer)**

| Inductance (Note 1) | Pulse Engineering                     | Valor Electronics                       | FEE Fil-Mag (Note 2) | PCA Electronics          |
|---------------------|---------------------------------------|-----------------------------------------|----------------------|--------------------------|
| 50 $\mu$ H          | 64101 (500V) (Note 4)<br>64106 (2 kV) | LT6001 (500V) (Note 4)<br>LT6031 (2 kV) |                      | EP9531-4 (2 kV) (Note 3) |
| 75 $\mu$ H          | 64102 (500V)<br>64107 (2 kV)          | LT6002 (500V)<br>LT6032 (2 kV)          | 23Z90                | EP9531-5 (2 kV)          |
| 100 $\mu$ H         | 64103 (500V)<br>64108 (2 kV)          | LT6003 (500V)<br>LT6033 (2 kV)          | 23Z91                | EP9531-6 (2 kV)          |
| 150 $\mu$ H         | 64104 (500V)<br>64109 (2 kV)          | LT6004 (500V)<br>LT6034 (2 kV)          | 23Z92                | EP9531-8 (2 kV)          |
| 250 $\mu$ H         |                                       | LT6005 (500V)<br>LT6035 (2 kV)          |                      | EP9531-11 (2 kV)         |

**Note 1:** Generally inductances range from 35  $\mu$ H to 300  $\mu$ H, this table only shows the more commonly used values.

**Note 2:** Information provided only listed 2 kV isolation components. To order surface mount add an "SM" to the part number.

**Note 3:** To specify 500V isolation add an "X" at the end of the part number.

**Note 4:** Surface mount versions also available.

**TABLE III. DC to DC Converters for the DP8392  
(-9V Output, ≥ 200 mA)**

|                 | 500V Isolation                                     |                                                    | 2000V Isolation                        |                                        |
|-----------------|----------------------------------------------------|----------------------------------------------------|----------------------------------------|----------------------------------------|
|                 | + 5V Input                                         | + 12V Input                                        | + 5V Input                             | + 12V Input                            |
| Valor           | PM7102                                             | PM7104                                             | PM6022<br>PM6045 (Note 1)              | PM6028<br>PM6030 (Note 1)              |
| (Switched)      |                                                    |                                                    | PM6044/PM6042                          | PM6079/PM6077                          |
| PCA Electronics | EPC1000P (Note 1)<br>EPC1015P (Note 1)<br>EPC1007P | EPC1005P (Note 1)<br>EPC1013P (Note 1)<br>EPC1008P | EPC1000H (Note 1)<br>EPC1015H (Note 1) | EPC1005H (Note 1)<br>EPC1013H (Note 1) |
| (Switched)      | EPC1002P (Note 1)                                  |                                                    | EPC1002H (Note 1)                      |                                        |

Note 1: These DC to DC Converters have a regulated output.

**TABLE IV. Pulse Transformer Vendors**

|   | Company and Address                                                               | Phone        | FAX          |
|---|-----------------------------------------------------------------------------------|--------------|--------------|
| 1 | <b>Pulse Engineering</b><br>P.O. Box 12235<br>San Diego, CA 92112                 | 619-674-8100 | 619-674-8262 |
| 2 | <b>Valor Electronics</b><br>9715 Business Park Avenue<br>San Diego, CA 92131-1642 | 619-537-2500 | 619-537-2525 |
| 3 | <b>FEE Fil-Mag</b><br>9445 Farnham<br>San Diego, CA 92123                         | 619-569-6577 | 619-569-6073 |
| 4 | <b>PCA Electronics</b><br>16799 Schoenborn St.<br>Sepulveda, CA 91343             | 818-892-0761 | 818-894-5791 |





## Section 3

### ETHERNET PROTOCOL PRODUCTS

#### **10 Mb/s Repeater Interface Controller Products**



### Section 3 Contents

|                                                                                    |       |
|------------------------------------------------------------------------------------|-------|
| DP83950B RIC Repeater Interface Controller .....                                   | 3-3   |
| DP83952 Repeater Interface Controller with Security Features (RIC II) .....        | 3-82  |
| DP83955A/DP83956A LERIC LitE Repeater Interface Controller .....                   | 3-171 |
| DP83957 10 Mb/s Repeater Information Base .....                                    | 3-222 |
| AN-781 DP83950EB-AT IEEE 802.3 Multi-Port Repeater Evaluation Kit .....            | 3-245 |
| AN-782 RIC-SONIC Interface .....                                                   | 3-258 |
| AN-783 DP83950 Twisted Pair Parametric Evaluation .....                            | 3-262 |
| AN-843 Introduction to Repeaters, the RIC and LERIC, and Their Applications .....  | 3-270 |
| AN-854 DP83956EB-AT LERIC (LitE Repeater Interface Controller) PC-AT Adapter ..... | 3-289 |
| AN-888 RFI Suppression Techniques in DP83950 RIC Based Systems .....               | 3-308 |
| AN-896 DP83956EB-SA Stand Alone Hub .....                                          | 3-319 |

# DP83950B RIC™

## Repeater Interface Controller

### General Description

The DP83950B Repeater Interface Controller "RIC" may be used to implement an IEEE 802.3 multiport repeater unit. It fully satisfies the IEEE 802.3 repeater specification including the functions defined by the repeater, segment partition and jabber lockup protection state machines.

The RIC has an on-chip phase-locked-loop (PLL) for Manchester data decoding, a Manchester encoder and an Elasticity Buffer for preamble regeneration.

Each RIC can connect to 13 cable segments via its network interface ports. One port is fully AUI compatible and is able to connect to an external MAU using the maximum length of AUI cable. The other 12 ports have integrated 10BASE-T transceivers. These transceiver functions may be bypassed so that the RIC may be used with external transceivers, for example DP8392 coaxial transceivers. In addition, large repeater units, containing several hundred ports may be constructed by cascading RICs together over an Inter-RIC bus.

The RIC is configurable for specific applications. It provides port status information for LED array displays and a simple interface for system processors. The RIC possesses multi-function counter and status flag arrays to facilitate network statistics gathering. A serial interface, known as the Management Interface is available for the collection of data in Managed Hub applications.

### Features

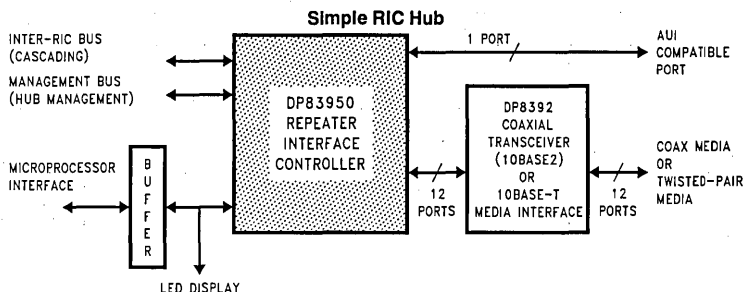
- Compliant with the IEEE 802.3 Repeater Specification
- 13 network connections (ports) per chip
- Selectable on-chip twisted-pair transceivers
- Cascadable for large hub applications
- Compatible with AUI compliant transceivers
- On-chip Elasticity Buffer, Manchester encoder and decoder

- Separate partition state machines for each port
- Provides port status information for LED displays including: receive, collision, partition and link status
- Power-up configuration options: Repeater and Partition Specifications, Transceiver Interface, Status Display, Processor Operations
- Simple processor interface for repeater management and port disable
- On-chip Event Counters and Event Flag Arrays
- Serial Management Interface to combine packet and repeater status information together
- CMOS process for low power dissipation
- Single 5V supply

### Table of Contents

- 1.0 SYSTEM DIAGRAM
- 2.0 CONNECTION DIAGRAM
- 3.0 PIN DESCRIPTIONS
- 4.0 BLOCK DIAGRAM
- 5.0 FUNCTIONAL DESCRIPTION
- 6.0 HUB MANAGEMENT SUPPORT
- 7.0 PORT LOGIC FUNCTIONS
- 8.0 RIC REGISTER DESCRIPTIONS
- 9.0 AC AND DC SPECIFICATIONS
- 10.0 AC TIMING TEST CONDITIONS
- 11.0 PHYSICAL DIMENSIONS

## 1.0 System Diagram



TL/F/11096-1

## 2.0 Connection Diagram—160 Pin PQFP Package

Pin Table (12 T.P. Ports + 1 AUI Bottom View)

| Pin Name        | Pin No. |
|-----------------|---------|
| TXO12P-         | 40      |
| TXO12+          | 39      |
| TXO12-          | 38      |
| TXO12P+         | 37      |
| RXI12-          | 36      |
| RXI12+          | 35      |
| V <sub>CC</sub> | 34      |
| GND             | 33      |
| RXI11-          | 32      |
| RXI11+          | 31      |
| TXO11P+         | 30      |
| TXO11-          | 29      |
| TXO11+          | 28      |
| TXO11P-         | 27      |
| V <sub>CC</sub> | 26      |
| GND             | 25      |
| TXO10P-         | 24      |
| TXO10+          | 23      |
| TXO10-          | 22      |
| TXO10P+         | 21      |
| RXI10-          | 20      |
| RXI10+          | 19      |
| V <sub>CC</sub> | 18      |
| GND             | 17      |
| RXI9-           | 16      |
| RXI9+           | 15      |
| TXO9P+          | 14      |
| TXO9-           | 13      |
| TXO9+           | 12      |
| TXO9P-          | 11      |
| V <sub>CC</sub> | 10      |
| GND             | 9       |
| TXO8P-          | 8       |
| TXO8+           | 7       |
| TXO8-           | 6       |
| TXO8P+          | 5       |
| RXI8-           | 4       |
| RXI8+           | 3       |
| V <sub>CC</sub> | 2       |
| GND             | 1       |

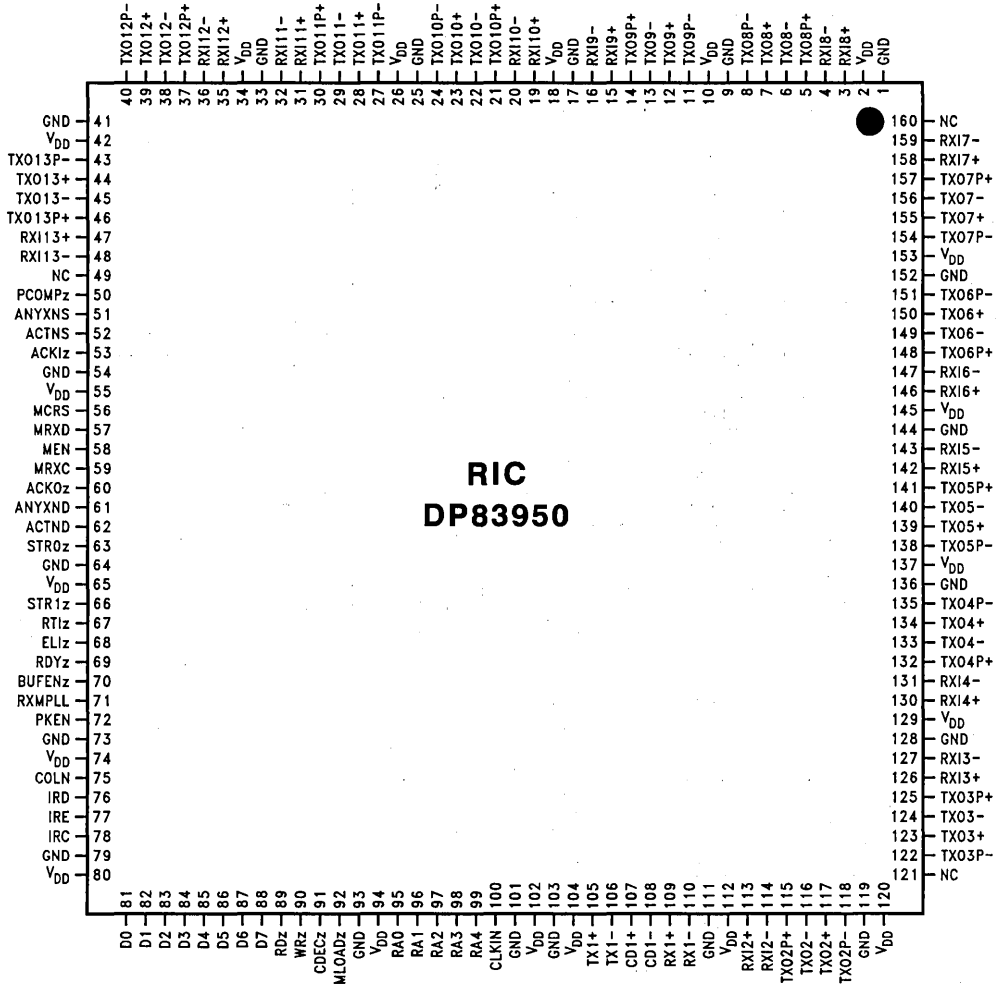
| Pin Name        | Pin No. |
|-----------------|---------|
| NC              | 160     |
| RXI7-           | 159     |
| RXI7+           | 158     |
| TXO7P+          | 157     |
| TXO7-           | 156     |
| TXO7+           | 155     |
| TXO7P-          | 154     |
| V <sub>CC</sub> | 153     |
| GND             | 152     |
| TXO6P-          | 151     |
| TXO6+           | 150     |
| TXO6-           | 149     |
| TXO6P+          | 148     |
| RXI6-           | 147     |
| RXI6+           | 146     |
| V <sub>CC</sub> | 145     |
| GND             | 144     |
| RXI5-           | 143     |
| RXI5+           | 142     |
| TXO5P+          | 141     |
| TXO5-           | 140     |
| TXO5+           | 139     |
| TXO5P-          | 138     |
| V <sub>CC</sub> | 137     |
| GND             | 136     |
| TXO4P-          | 135     |
| TXO4+           | 134     |
| TXO4-           | 133     |
| TXO4P+          | 132     |
| RXI4-           | 131     |
| RXI4+           | 130     |
| V <sub>CC</sub> | 129     |
| GND             | 128     |
| RXI3-           | 127     |
| RXI3+           | 126     |
| TXO3P+          | 125     |
| TXO3-           | 124     |
| TXO3+           | 123     |
| TXO3P-          | 122     |
| NC              | 121     |

| Pin Name           | Pin No. |
|--------------------|---------|
| V <sub>CC</sub>    | 120     |
| GND                | 119     |
| TXO2P-             | 118     |
| TXO2+              | 117     |
| TXO2-              | 116     |
| TXO2P+             | 115     |
| RXI2-              | 114     |
| RXI2+              | 113     |
| V <sub>CC</sub>    | 112     |
| GND                | 111     |
| RX1-               | 110     |
| RX1+               | 109     |
| CD1-               | 108     |
| CD1+               | 107     |
| TX1-               | 106     |
| TX1+               | 105     |
| V <sub>CC</sub>    | 104     |
| GND                | 103     |
| V <sub>CC</sub>    | 102     |
| GND                | 101     |
| CLKIN              | 100     |
| RA4                | 99      |
| RA3                | 98      |
| RA2                | 97      |
| RA1                | 96      |
| RA0                | 95      |
| V <sub>CC</sub>    | 94      |
| GND                | 93      |
| $\overline{MLOAD}$ | 92      |
| $\overline{CDEC}$  | 91      |
| $\overline{WR}$    | 90      |
| $\overline{RD}$    | 89      |
| D7                 | 88      |
| D6                 | 87      |
| D5                 | 86      |
| D4                 | 85      |
| D3                 | 84      |
| D2                 | 83      |
| D1                 | 82      |
| D0                 | 81      |

| Pin Name           | Pin No. |
|--------------------|---------|
| V <sub>CC</sub>    | 80      |
| GND                | 79      |
| IRC                | 78      |
| IRE                | 77      |
| IRD                | 76      |
| COLN               | 75      |
| V <sub>CC</sub>    | 74      |
| GND                | 73      |
| PKEN               | 72      |
| RXMPLL             | 71      |
| $\overline{BUFEN}$ | 70      |
| $\overline{RDY}$   | 69      |
| $\overline{ELI}$   | 68      |
| $\overline{RTI}$   | 67      |
| $\overline{STR1}$  | 66      |
| V <sub>CC</sub>    | 65      |
| GND                | 64      |
| $\overline{STR0}$  | 63      |
| ACTND              | 62      |
| ANYXND             | 61      |
| $\overline{ACK0}$  | 60      |
| MRXC               | 59      |
| MEN                | 58      |
| MRXD               | 57      |
| MCRS               | 56      |
| V <sub>CC</sub>    | 55      |
| GND                | 54      |
| $\overline{ACK1}$  | 53      |
| ACTNS              | 52      |
| ANYXNS             | 51      |
| $\overline{PCOMP}$ | 50      |
| NC                 | 49      |
| RXI13-             | 48      |
| RXI13+             | 47      |
| TXO13P+            | 46      |
| TXO13-             | 45      |
| TXO13+             | 44      |
| TXO13P-            | 43      |
| V <sub>CC</sub>    | 42      |
| GND                | 41      |

Note: NC = No Connect

2.0 Connection Diagram—160 Pin PQFP Package (Continued)



**RIC**  
**DP83950**

Ports 2-13 TP  
Port 1 AUI

Order Number DP83950BVQB  
See NS Package Number VUL160A

TL/F/11096-42

## 2.0 Connection Diagram—160 Pin PQFP Package (Continued)

Pin Table (1–5 AUI + 6–13 T.P. Ports)

| Pin Name        | Pin No. |
|-----------------|---------|
| TXO12P–         | 40      |
| TXO12+          | 39      |
| TXO12–          | 38      |
| TXO12P+         | 37      |
| RXI12–          | 36      |
| RXI12+          | 35      |
| V <sub>CC</sub> | 34      |
| GND             | 33      |
| RXI11–          | 32      |
| RXI11+          | 31      |
| TXO11P+         | 30      |
| TXO11–          | 29      |
| TXO11+          | 28      |
| TXO11P–         | 27      |
| V <sub>CC</sub> | 26      |
| GND             | 25      |
| TXO10P–         | 24      |
| TXO10+          | 23      |
| TXO10–          | 22      |
| TXO10P+         | 21      |
| RXI10–          | 20      |
| RXI10+          | 19      |
| V <sub>CC</sub> | 18      |
| GND             | 17      |
| RXI9–           | 16      |
| RXI9+           | 15      |
| TXO9P+          | 14      |
| TXO9–           | 13      |
| TXO9+           | 12      |
| TXO9P–          | 11      |
| V <sub>CC</sub> | 10      |
| GND             | 9       |
| TXO8P–          | 8       |
| TXO8+           | 7       |
| TXO8–           | 6       |
| TXO8P+          | 5       |
| RXI8–           | 4       |
| RXI8+           | 3       |
| V <sub>CC</sub> | 2       |
| GND             | 1       |

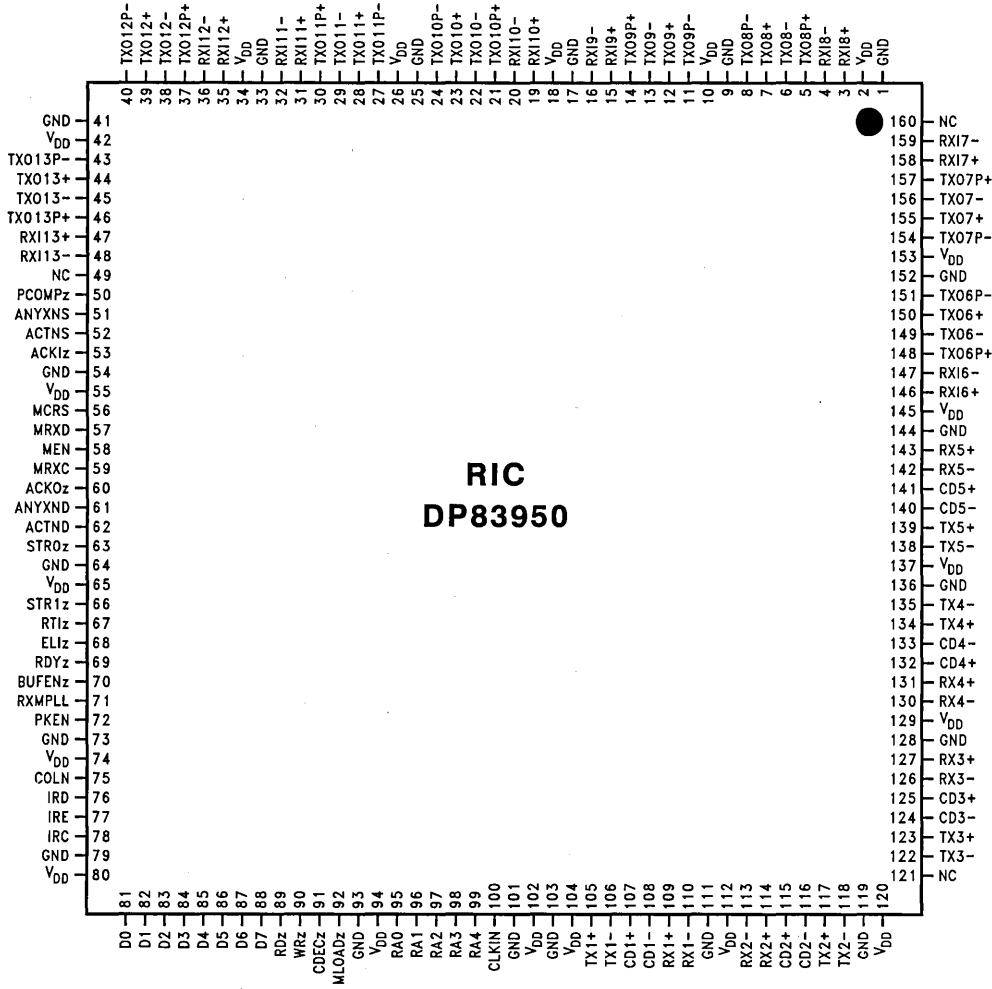
| Pin Name        | Pin No. |
|-----------------|---------|
| NC              | 160     |
| RXI7–           | 159     |
| RXI7+           | 158     |
| TXO7P+          | 157     |
| TXO7–           | 156     |
| TXO7+           | 155     |
| TXO7P–          | 154     |
| V <sub>CC</sub> | 153     |
| GND             | 152     |
| TXO6P–          | 151     |
| TXO6+           | 150     |
| TXO6–           | 149     |
| TXO6P+          | 148     |
| RXI6–           | 147     |
| RXI6+           | 146     |
| V <sub>CC</sub> | 145     |
| GND             | 144     |
| RX5+            | 143     |
| RX5–            | 142     |
| CD5+            | 141     |
| CD5–            | 140     |
| TX5+            | 139     |
| TX5–            | 138     |
| V <sub>CC</sub> | 137     |
| GND             | 136     |
| TX4–            | 135     |
| TX4+            | 134     |
| CD4–            | 133     |
| CD4+            | 132     |
| RX4+            | 131     |
| RX4–            | 130     |
| V <sub>CC</sub> | 129     |
| GND             | 128     |
| RX3+            | 127     |
| RX3–            | 126     |
| CD3+            | 125     |
| CD3–            | 124     |
| TX3+            | 123     |
| TX3–            | 122     |
| NC              | 121     |

| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | 120     |
| GND             | 119     |
| TX2–            | 118     |
| TX2+            | 117     |
| CD2–            | 116     |
| CD2+            | 115     |
| RX2+            | 114     |
| RX2–            | 113     |
| V <sub>CC</sub> | 112     |
| GND             | 111     |
| RX1–            | 110     |
| RX1+            | 109     |
| CD1–            | 108     |
| CD1+            | 107     |
| TX1–            | 106     |
| TX1+            | 105     |
| V <sub>CC</sub> | 104     |
| GND             | 103     |
| V <sub>CC</sub> | 102     |
| GND             | 101     |
| CLKIN           | 100     |
| RA4             | 99      |
| RA3             | 98      |
| RA2             | 97      |
| RA1             | 96      |
| RA0             | 95      |
| V <sub>CC</sub> | 94      |
| GND             | 93      |
| MLOAD           | 92      |
| CDEC            | 91      |
| WR              | 90      |
| RD              | 89      |
| D7              | 88      |
| D6              | 87      |
| D5              | 86      |
| D4              | 85      |
| D3              | 84      |
| D2              | 83      |
| D1              | 82      |
| D0              | 81      |

| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | 80      |
| GND             | 79      |
| IRC             | 78      |
| IRE             | 77      |
| IRD             | 76      |
| COLN            | 75      |
| V <sub>CC</sub> | 74      |
| GND             | 73      |
| PKEN            | 72      |
| RXMPLL          | 71      |
| BUFEN           | 70      |
| RDY             | 69      |
| ELI             | 68      |
| RTI             | 67      |
| STR1            | 66      |
| V <sub>CC</sub> | 65      |
| GND             | 64      |
| STR0            | 63      |
| ACTND           | 62      |
| ANYXND          | 61      |
| ACKO            | 60      |
| MRXC            | 59      |
| MEN             | 58      |
| MRXD            | 57      |
| MCRS            | 56      |
| V <sub>CC</sub> | 55      |
| GND             | 54      |
| ACKI            | 53      |
| ACTNS           | 52      |
| ANYXNS          | 51      |
| PCOMP           | 50      |
| NC              | 49      |
| RXI13–          | 48      |
| RXI13+          | 47      |
| TXO13P+         | 46      |
| TXO13–          | 45      |
| TXO13+          | 44      |
| TXO13P–         | 43      |
| V <sub>CC</sub> | 42      |
| GND             | 41      |

Note: NC = No Connect

2.0 Connection Diagram—160 Pin PQFP Package (Continued)



Ports 6–13 TP  
Ports 1–5 AUI  
  
Order Number DP83950BVQB  
See NS Package Number VUL160A

TL/F/11096-43

## 2.0 Connection Diagram—160 Pin PQFP Package (Continued)

Pin Table (1–7 AUI + 8–13 T.P. Ports)

| Pin Name        | Pin No. |
|-----------------|---------|
| TXO12P–         | 40      |
| TXO12+          | 39      |
| TXO12–          | 38      |
| TXO12P+         | 37      |
| RXI12–          | 36      |
| RXI12+          | 35      |
| V <sub>CC</sub> | 34      |
| GND             | 33      |
| RXI11–          | 32      |
| RXI11+          | 31      |
| TXO11P+         | 30      |
| TXO11–          | 29      |
| TXO11+          | 28      |
| TXO11P–         | 27      |
| V <sub>CC</sub> | 26      |
| GND             | 25      |
| TXO10P–         | 24      |
| TXO10+          | 23      |
| TXO10–          | 22      |
| TXO10P+         | 21      |
| RXI10–          | 20      |
| RXI10+          | 19      |
| V <sub>CC</sub> | 18      |
| GND             | 17      |
| RXI9–           | 16      |
| RXI9+           | 15      |
| TXO9P+          | 14      |
| TXO9–           | 13      |
| TXO9+           | 12      |
| TXO9P–          | 11      |
| V <sub>CC</sub> | 10      |
| GND             | 9       |
| TXO8P–          | 8       |
| TXO8+           | 7       |
| TXO8–           | 6       |
| TXO8P+          | 5       |
| RXI8–           | 4       |
| RXI8+           | 3       |
| V <sub>CC</sub> | 2       |
| GND             | 1       |

| Pin Name        | Pin No. |
|-----------------|---------|
| NC              | 160     |
| RX7+            | 159     |
| RX7–            | 158     |
| CD7+            | 157     |
| CD7–            | 156     |
| TX7+            | 155     |
| TX7–            | 154     |
| V <sub>CC</sub> | 153     |
| GND             | 152     |
| TX6–            | 151     |
| TX6+            | 150     |
| CD6–            | 149     |
| CD6+            | 148     |
| RX6+            | 147     |
| RX6–            | 146     |
| V <sub>CC</sub> | 145     |
| GND             | 144     |
| RX5+            | 143     |
| RX5–            | 142     |
| CD5+            | 141     |
| CD5–            | 140     |
| TX5+            | 139     |
| TX5–            | 138     |
| V <sub>CC</sub> | 137     |
| GND             | 136     |
| TX4–            | 135     |
| TX4+            | 134     |
| CD4–            | 133     |
| CD4+            | 132     |
| RX4+            | 131     |
| RX4–            | 130     |
| V <sub>CC</sub> | 129     |
| GND             | 128     |
| RX3+            | 127     |
| RX3–            | 126     |
| CD3+            | 125     |
| CD3–            | 124     |
| TX3+            | 123     |
| TX3–            | 122     |
| NC              | 121     |

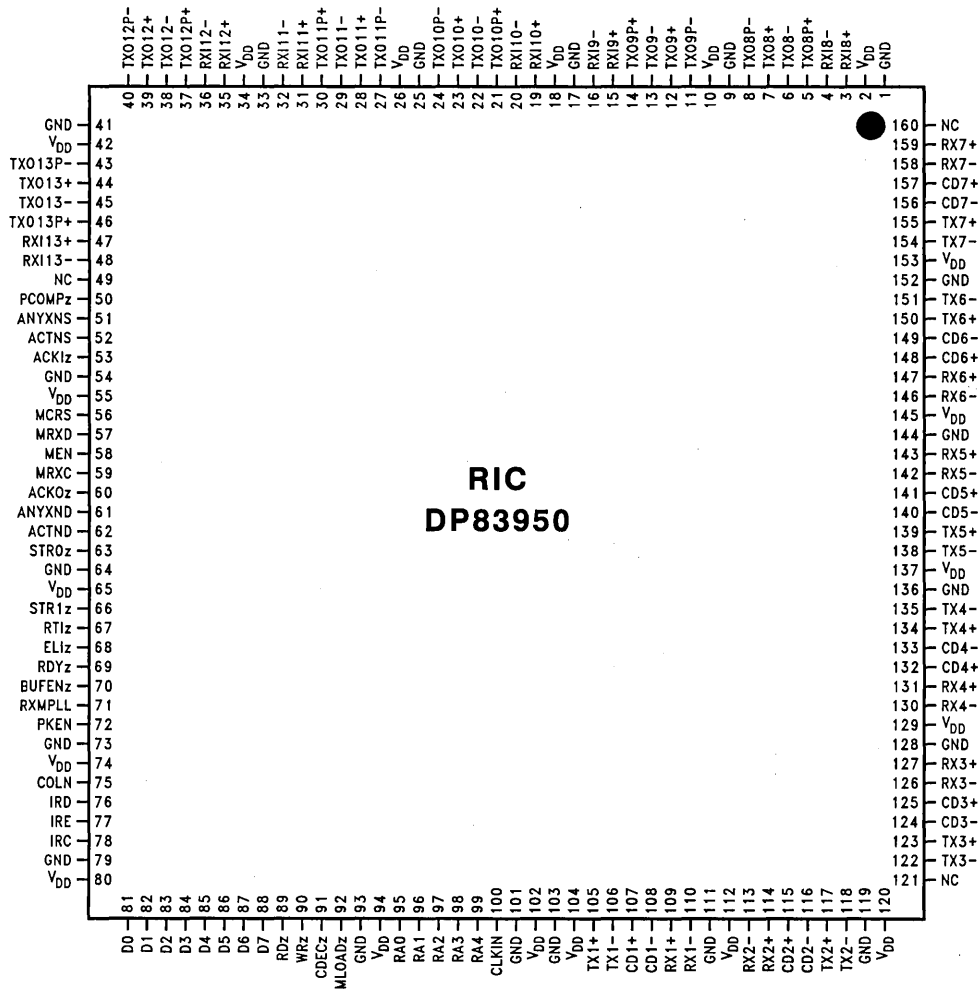
| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | 120     |
| GND             | 119     |
| TX2–            | 118     |
| TX2+            | 117     |
| CD2–            | 116     |
| CD2+            | 115     |
| RX2+            | 114     |
| RX2–            | 113     |
| V <sub>CC</sub> | 112     |
| GND             | 111     |
| RX1–            | 110     |
| RX1+            | 109     |
| CD1–            | 108     |
| CD1+            | 107     |
| TX1–            | 106     |
| TX1+            | 105     |
| V <sub>CC</sub> | 104     |
| GND             | 103     |
| V <sub>CC</sub> | 102     |
| GND             | 101     |
| CLKIN           | 100     |
| RA4             | 99      |
| RA3             | 98      |
| RA2             | 97      |
| RA1             | 96      |
| RA0             | 95      |
| V <sub>CC</sub> | 94      |
| GND             | 93      |
| MLOAD           | 92      |
| CDEC            | 91      |
| WR              | 90      |
| RD              | 89      |
| D7              | 88      |
| D6              | 87      |
| D5              | 86      |
| D4              | 85      |
| D3              | 84      |
| D2              | 83      |
| D1              | 82      |
| D0              | 81      |

| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | 80      |
| GND             | 79      |
| IRC             | 78      |
| IRE             | 77      |
| IRD             | 76      |
| COLN            | 75      |
| V <sub>CC</sub> | 74      |
| GND             | 73      |
| PKEN            | 72      |
| RXMPLL          | 71      |
| BUFEN           | 70      |
| RDY             | 69      |
| ELI             | 68      |
| RTI             | 67      |
| STR1            | 66      |
| V <sub>CC</sub> | 65      |
| GND             | 64      |
| STR0            | 63      |
| ACTND           | 62      |
| ANYXND          | 61      |
| ACK0            | 60      |
| MRXC            | 59      |
| MEN             | 58      |
| MRXD            | 57      |
| MCRS            | 56      |
| V <sub>CC</sub> | 55      |
| GND             | 54      |
| ACK1            | 53      |
| ACTNS           | 52      |
| ANYXNS          | 51      |
| PCOMP           | 50      |
| NC              | 49      |
| RXI13–          | 48      |
| RXI13+          | 47      |
| TXO13P+         | 46      |
| TXO13–          | 45      |
| TXO13+          | 44      |
| TXO13P–         | 43      |
| V <sub>CC</sub> | 42      |
| GND             | 41      |

Note: NC = No Connect



## 2.0 Connection Diagram—160 Pin PQFP Package (Continued)



TL/F/11096-44

Ports 8-13 TP

Ports 1-7 AUI

Order Number DP83950BVQB  
See NS Package Number VUL160A

## 2.0 Connection Diagram—160 Pin PQFP Package (Continued)

Pin Table (All AUI Ports)

| Pin Name        | Pin No. |
|-----------------|---------|
| TX12-           | 40      |
| TX12+           | 39      |
| CD12-           | 38      |
| CD12+           | 37      |
| RX12+           | 36      |
| RX12-           | 35      |
| V <sub>CC</sub> | 34      |
| GND             | 33      |
| RX11+           | 32      |
| RX11-           | 31      |
| CD11+           | 30      |
| CD11-           | 29      |
| TX11+           | 28      |
| TX11-           | 27      |
| V <sub>CC</sub> | 26      |
| GND             | 25      |
| TX10-           | 24      |
| TX10+           | 23      |
| CD10-           | 22      |
| CD10+           | 21      |
| RX10+           | 20      |
| RX10-           | 19      |
| V <sub>CC</sub> | 18      |
| GND             | 17      |
| RX9+            | 16      |
| RX9-            | 15      |
| CD9+            | 14      |
| CD9-            | 13      |
| TX9+            | 12      |
| TX9-            | 11      |
| V <sub>CC</sub> | 10      |
| GND             | 9       |
| TX8-            | 8       |
| TX8+            | 7       |
| CD8-            | 6       |
| CD8+            | 5       |
| RX8+            | 4       |
| RX8-            | 3       |
| V <sub>CC</sub> | 2       |
| GND             | 1       |

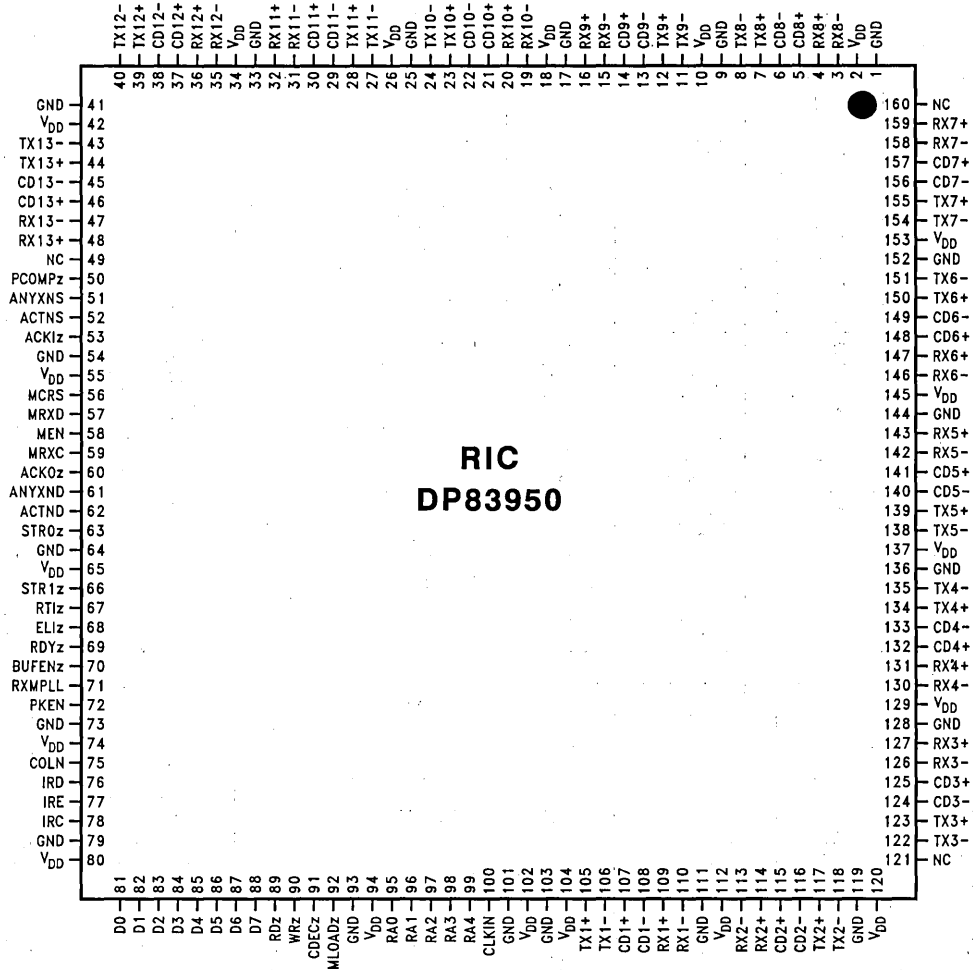
| Pin Name        | Pin No. |
|-----------------|---------|
| NC              | 160     |
| RX7+            | 159     |
| RX7-            | 158     |
| CD7+            | 157     |
| CD7-            | 156     |
| TX7+            | 155     |
| TX7-            | 154     |
| V <sub>CC</sub> | 153     |
| GND             | 152     |
| TX6-            | 151     |
| TX6+            | 150     |
| CD6-            | 149     |
| CD6+            | 148     |
| RX6+            | 147     |
| RX6-            | 146     |
| V <sub>CC</sub> | 145     |
| GND             | 144     |
| RX5+            | 143     |
| RX5-            | 142     |
| CD5+            | 141     |
| CD5-            | 140     |
| TX5+            | 139     |
| TX5-            | 138     |
| V <sub>CC</sub> | 137     |
| GND             | 136     |
| TX4-            | 135     |
| TX4+            | 134     |
| CD4-            | 133     |
| CD4+            | 132     |
| RX4+            | 131     |
| RX4-            | 130     |
| V <sub>CC</sub> | 129     |
| GND             | 128     |
| RX3+            | 127     |
| RX3-            | 126     |
| CD3+            | 125     |
| CD3-            | 124     |
| TX3+            | 123     |
| TX3-            | 122     |
| NC              | 121     |

| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | 120     |
| GND             | 119     |
| TX2-            | 118     |
| TX2+            | 117     |
| CD2-            | 116     |
| CD2+            | 115     |
| RX2+            | 114     |
| RX2-            | 113     |
| V <sub>CC</sub> | 112     |
| GND             | 111     |
| RX1-            | 110     |
| RX1+            | 109     |
| CD1-            | 108     |
| CD1+            | 107     |
| TX1-            | 106     |
| TX1+            | 105     |
| V <sub>CC</sub> | 104     |
| GND             | 103     |
| V <sub>CC</sub> | 102     |
| GND             | 101     |
| CLKIN           | 100     |
| RA4             | 99      |
| RA3             | 98      |
| RA2             | 97      |
| RA1             | 96      |
| RA0             | 95      |
| V <sub>CC</sub> | 94      |
| GND             | 93      |
| MLOAD           | 92      |
| CDEC            | 91      |
| WR              | 90      |
| RD              | 89      |
| D7              | 88      |
| D6              | 87      |
| D5              | 86      |
| D4              | 85      |
| D3              | 84      |
| D2              | 83      |
| D1              | 82      |
| D0              | 81      |

| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | 80      |
| GND             | 79      |
| IRC             | 78      |
| IRE             | 77      |
| IRD             | 76      |
| COLN            | 75      |
| V <sub>CC</sub> | 74      |
| GND             | 73      |
| PKEN            | 72      |
| RXMPLL          | 71      |
| BUFEN           | 70      |
| RDY             | 69      |
| ELI             | 68      |
| RTI             | 67      |
| STR1            | 66      |
| V <sub>CC</sub> | 65      |
| GND             | 64      |
| STR0            | 63      |
| ACTND           | 62      |
| ANYXND          | 61      |
| ACKO            | 60      |
| MRXC            | 59      |
| MEN             | 58      |
| MRXD            | 57      |
| MCRS            | 56      |
| V <sub>CC</sub> | 55      |
| GND             | 54      |
| ACKI            | 53      |
| ACTNS           | 52      |
| ANYXNS          | 51      |
| PCOMP           | 50      |
| NC              | 49      |
| RX13+           | 48      |
| RX13-           | 47      |
| CD13+           | 46      |
| CD13-           | 45      |
| TX13+           | 44      |
| TX13-           | 43      |
| V <sub>CC</sub> | 42      |
| GND             | 41      |

Note: NC = No Connect

2.0 Connection Diagram—160 Pin PQFP Package (Continued)



TL/F/11096-45

All AUI Ports

Order Number DP83950BVQB  
See NS Package Number VUL160A

## 2.0 Connection Diagram—160 Pin PGA Package (Continued)

Pin Table (12 T.P. Ports + 1 AUI Bottom View)

| Pin Name        | Pin No. |
|-----------------|---------|
| TXO12P-         | A15     |
| TXO12+          | A14     |
| TXO12-          | B14     |
| TXO12P+         | C13     |
| RXI12-          | B13     |
| RXI12+          | A13     |
| V <sub>CC</sub> | C12     |
| GND             | C11     |
| RXI11-          | B12     |
| RXI11+          | B11     |
| TXO11P+         | A12     |
| TXO11-          | A11     |
| TXO11+          | C10     |
| TXO11P-         | A10     |
| V <sub>CC</sub> | B10     |
| GND             | B9      |
| TXO10P-         | C9      |
| TXO10+          | C8      |
| TXO10-          | A9      |
| TXO10P+         | A8      |
| RXI10-          | B8      |
| RXI10+          | B7      |
| V <sub>CC</sub> | C7      |
| GND             | A7      |
| RXI9-           | A6      |
| RXI9+           | B6      |
| TXO9P+          | C6      |
| TXO9-           | C5      |
| TXO9+           | B5      |
| TXO9P-          | A5      |
| V <sub>CC</sub> | A4      |
| GND             | B4      |
| TXO8P-          | C4      |
| TXO8+           | A3      |
| TXO8-           | C3      |
| TXO8P+          | D4      |
| RXI8-           | B3      |
| RXI8+           | B2      |
| V <sub>CC</sub> | A2      |
| GND             | D3      |

| Pin Name        | Pin No. |
|-----------------|---------|
| RXI7-           | C2      |
| RXI7+           | A1      |
| TXO7P+          | B1      |
| TXO7-           | D2      |
| TXO7+           | E3      |
| TXO7P-          | F3      |
| V <sub>CC</sub> | C1      |
| GND             | D1      |
| TXO6P-          | E2      |
| TXO6+           | G3      |
| TXO6-           | F2      |
| TXO6P+          | E1      |
| RXI6-           | G2      |
| RXI6+           | H3      |
| NC              | F1      |
| NC              | G1      |
| V <sub>CC</sub> | H2      |
| GND             | J3      |
| RXI5-           | J2      |
| RXI5+           | H1      |
| TXO5P+          | J1      |
| TXO5-           | K1      |
| TXO5+           | K3      |
| TXO5P-          | K2      |
| V <sub>CC</sub> | L1      |
| GND             | L2      |
| TXO4P-          | M1      |
| TXO4+           | L3      |
| TXO4-           | M2      |
| TXO4P+          | N1      |
| RXI4-           | N2      |
| RXI4+           | M3      |
| V <sub>CC</sub> | P1      |
| GND             | R1      |
| RXI3-           | P2      |
| RXI3+           | N3      |
| TXO3P+          | P3      |
| TXO3-           | R2      |
| TXO3+           | N4      |
| TXO3P-          | R3      |

| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | S1      |
| GND             | P4      |
| TXO2P-          | S2      |
| TXO2+           | S3      |
| TXO2-           | R4      |
| TXO2P+          | P5      |
| RXI2-           | R5      |
| RXI2+           | S4      |
| V <sub>CC</sub> | S5      |
| GND             | S6      |
| RX1-            | P6      |
| RX1+            | R6      |
| CD1-            | S7      |
| CD1+            | R7      |
| TX1-            | P7      |
| TX1+            | P8      |
| V <sub>CC</sub> | R8      |
| GND             | S8      |
| V <sub>CC</sub> | S9      |
| GND             | R9      |
| CLKIN           | P9      |
| RA4             | S10     |
| RA3             | R10     |
| RA2             | S11     |
| RA1             | P10     |
| RA0             | R11     |
| V <sub>CC</sub> | S12     |
| GND             | R12     |
| MLOAD           | P11     |
| CDEC            | S13     |
| WR              | R13     |
| RD              | S14     |
| D7              | P12     |
| D6              | R14     |
| D5              | S15     |
| D4              | P13     |
| D3              | P14     |
| D2              | R15     |
| D1              | S16     |
| D0              | R16     |

| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | N13     |
| GND             | P15     |
| IRC             | N14     |
| IRE             | P16     |
| IRD             | N15     |
| COLN            | N16     |
| V <sub>CC</sub> | M15     |
| GND             | M14     |
| PKEN            | L14     |
| RXM             | L15     |
| BUFEN           | M16     |
| RDY             | L16     |
| ELI             | K16     |
| RTI             | K14     |
| STR1            | K15     |
| V <sub>CC</sub> | J16     |
| GND             | J15     |
| STR0            | J14     |
| ACTND           | H16     |
| ANYXND          | H15     |
| ACKO            | H14     |
| MRXC            | G14     |
| MEN             | G15     |
| MRXD            | G16     |
| MCRS            | F16     |
| V <sub>CC</sub> | F14     |
| GND             | F15     |
| ACKI            | E15     |
| ACTNS           | E14     |
| ANYXNS          | E16     |
| PCOMP           | D16     |
| RXI13-          | D15     |
| RXI13+          | D14     |
| TXO13P+         | C16     |
| TXO13-          | C15     |
| TXO13+          | B16     |
| TXO13P-         | B15     |
| V <sub>CC</sub> | D13     |
| GND             | C14     |

Note: NC = No Connect

2.0 Connection Diagram—160 Pin PGA Package (Continued)

|   |                 |                 |        |                 |                 |        |                 |                 |                 |                 |        |                 |                 |                 |                 |         |
|---|-----------------|-----------------|--------|-----------------|-----------------|--------|-----------------|-----------------|-----------------|-----------------|--------|-----------------|-----------------|-----------------|-----------------|---------|
| S | V <sub>CC</sub> | TX02P-          | TX02+  | RX12+           | V <sub>CC</sub> | GND    | CD1-            | GND             | V <sub>CC</sub> | RA4             | RA2    | V <sub>CC</sub> | CDEC            | RD              | D5              | D1      |
|   | 98              | 96              | 95     | 91              | 90              | 89     | 86              | 81              | 80              | 77              | 75     | 72              | 69              | 67              | 64              | 60      |
| R | GND             | TX03-           | TX03P- | TX02-           | RX12-           | RX1+   | CD1+            | V <sub>CC</sub> | GND             | RA3             | RA0    | GND             | WR              | D6              | D2              | D0      |
|   | 105             | 101             | 99     | 94              | 92              | 87     | 85              | 82              | 79              | 76              | 73     | 71              | 68              | 65              | 61              | 59      |
| P | V <sub>CC</sub> | RX13-           | TX03P+ | GND             | TX02P+          | RX1-   | TX1-            | TX1+            | CLKIN           | RA1             | MLOAD  | D7              | D4              | D3              | GND             | IRE     |
|   | 106             | 104             | 102    | 97              | 93              | 88     | 84              | 83              | 78              | 74              | 70     | 66              | 63              | 62              | 57              | 55      |
| N | TX04P+          | RX14-           | RX13+  | TX03+           |                 |        |                 |                 |                 |                 |        |                 | V <sub>CC</sub> | IRC             | IRD             | COLN    |
|   | 109             | 108             | 103    | 100             |                 |        |                 |                 |                 |                 |        |                 | 58              | 56              | 54              | 53      |
| M | TX04P-          | TX04-           | RX14+  |                 |                 |        |                 |                 |                 |                 |        |                 | GND             | V <sub>CC</sub> | BUFEN           |         |
|   | 112             | 110             | 107    |                 |                 |        |                 |                 |                 |                 |        |                 | 51              | 52              | 48              |         |
| L | V <sub>CC</sub> | GND             | TX04+  |                 |                 |        |                 |                 |                 |                 |        |                 | PKEN            | RXM             | RDY             |         |
|   | 114             | 113             | 111    |                 |                 |        |                 |                 |                 |                 |        |                 | 50              | 49              | 47              |         |
| K | TX05-           | TX05P-          | TX05+  |                 |                 |        |                 |                 |                 |                 |        |                 | RTI             | STR1            | ELI             |         |
|   | 117             | 115             | 116    |                 |                 |        |                 |                 |                 |                 |        |                 | 45              | 44              | 46              |         |
| J | TX05P+          | RX15-           | GND    |                 |                 |        |                 |                 |                 |                 |        |                 | STR0            | GND             | V <sub>CC</sub> |         |
|   | 118             | 120             | 121    |                 |                 |        |                 |                 |                 |                 |        |                 | 41              | 42              | 43              |         |
| H | RX15+           | V <sub>CC</sub> | RX16+  |                 |                 |        |                 |                 |                 |                 |        |                 | ACK0            | ANYXND          | ACTND           |         |
|   | 119             | 122             | 125    |                 |                 |        |                 |                 |                 |                 |        |                 | 38              | 39              | 40              |         |
| G | NC              | RX16-           | TX06+  |                 |                 |        |                 |                 |                 |                 |        |                 | MRXC            | MEN             | MRXD            |         |
|   | 123             | 126             | 129    |                 |                 |        |                 |                 |                 |                 |        |                 | 37              | 36              | 35              |         |
| F | NC              | TX06-           | TX07P- |                 |                 |        |                 |                 |                 |                 |        |                 | V <sub>CC</sub> | GND             | MCRS            |         |
|   | 124             | 128             | 133    |                 |                 |        |                 |                 |                 |                 |        |                 | 33              | 32              | 34              |         |
| E | TX06P+          | TX06P-          | TX07+  |                 |                 |        |                 |                 |                 |                 |        |                 | ACTNS           | ACK1            | ANYXNS          |         |
|   | 127             | 130             | 134    |                 |                 |        |                 |                 |                 |                 |        |                 | 30              | 31              | 29              |         |
| D | GND             | TX07-           | GND    | TX08P+          |                 |        |                 |                 |                 |                 |        |                 | V <sub>CC</sub> | RX113+          | RX113-          | PCOMP   |
|   | 131             | 135             | 139    | 143             |                 |        |                 |                 |                 |                 |        |                 | 21              | 26              | 27              | 28      |
| C | V <sub>CC</sub> | RX17-           | TX08-  | TX08P-          | TX09-           | TX09P+ | V <sub>CC</sub> | TX010+          | TX010P-         | TX011+          | GND    | V <sub>CC</sub> | TX012P+         | GND             | TX013-          | TX013P+ |
|   | 132             | 138             | 144    | 146             | 151             | 152    | 156             | 2               | 3               | 7               | 12     | 13              | 16              | 20              | 24              | 25      |
| B | TX07P+          | RX18+           | RX18-  | GND             | TX09+           | RX19+  | RX110+          | RX110-          | GND             | V <sub>CC</sub> | RX111+ | RX111-          | RX112-          | TX012-          | TX013P-         | TX013+  |
|   | 136             | 141             | 142    | 147             | 150             | 153    | 157             | 158             | 4               | 5               | 10     | 11              | 15              | 17              | 22              | 23      |
| A | RX17+           | V <sub>CC</sub> | TX08+  | V <sub>CC</sub> | TX09P-          | RX19-  | GND             | TX010P+         | TX010-          | TX011P-         | TX011- | TX011P+         | RX112+          | TX012+          | TX012P-         |         |
|   | 137             | 140             | 145    | 148             | 149             | 154    | 155             | 159             | 1               | 6               | 8      | 9               | 14              | 18              | 19              |         |
|   | 1               | 2               | 3      | 4               | 5               | 6      | 7               | 8               | 9               | 0               | 1      | 2               | 3               | 4               | 5               | 6       |

RIC  
DP83950

Bottom View  
1 AUI + 2-13 T.P. Ports

Order Number DP83950BNU  
See NS Package Number UP159A

TL/F/11096-2

## 2.0 Connection Diagram—160 Pin PGA Package (Continued)

Pin Table (1–5 AUI + 6–13 T.P. Ports)

| Pin Name        | Pin No. |
|-----------------|---------|
| TXO12P–         | A15     |
| TXO12+          | A14     |
| TXO12–          | B14     |
| TXO12P+         | C13     |
| RXI12–          | B13     |
| RXI12+          | A13     |
| V <sub>CC</sub> | C12     |
| GND             | C11     |
| RXI11–          | B12     |
| RXI11+          | B11     |
| TXO11P+         | A12     |
| TXO11–          | A11     |
| TXO11+          | C10     |
| TXO11P–         | A10     |
| V <sub>CC</sub> | B10     |
| GND             | B9      |
| TXO10P–         | C9      |
| TXO10+          | C8      |
| TXO10–          | A9      |
| TXO10P+         | A8      |
| RXI10–          | B8      |
| RXI10+          | B7      |
| V <sub>CC</sub> | C7      |
| GND             | A7      |
| RXI9–           | A6      |
| RXI9+           | B6      |
| TXO9P+          | C6      |
| TXO9–           | C5      |
| TXO9+           | B5      |
| TXO9P–          | A5      |
| V <sub>CC</sub> | A4      |
| GND             | B4      |
| TXO8P–          | C4      |
| TXO8+           | A3      |
| TXO8–           | C3      |
| TXO8P+          | D4      |
| RXI8–           | B3      |
| RXI8+           | B2      |
| V <sub>CC</sub> | A2      |
| GND             | D3      |

| Pin Name        | Pin No. |
|-----------------|---------|
| RXI7–           | C2      |
| RXI7+           | A1      |
| TXO7P+          | B1      |
| TXO7–           | D2      |
| TXO7+           | E3      |
| TXO7P–          | F3      |
| V <sub>CC</sub> | C1      |
| GND             | D1      |
| TXO6P–          | E2      |
| TXO6+           | G3      |
| TXO6–           | F2      |
| TXO6P+          | E1      |
| RXI6–           | G2      |
| RXI6+           | H3      |
| NC              | F1      |
| NC              | G1      |
| V <sub>CC</sub> | H2      |
| GND             | J3      |
| RX5+            | J2      |
| RX5–            | H1      |
| CD5+            | J1      |
| CD5–            | K1      |
| TX5+            | K3      |
| TX5–            | K2      |
| V <sub>CC</sub> | L1      |
| GND             | L2      |
| TX4–            | M1      |
| TX4+            | L3      |
| CD4–            | M2      |
| CD4+            | N1      |
| RX4+            | N2      |
| RX4–            | M3      |
| V <sub>CC</sub> | P1      |
| GND             | R1      |
| RX3+            | P2      |
| RX3–            | N3      |
| CD3+            | P3      |
| CD3–            | R2      |
| TX3+            | N4      |
| TX3–            | R3      |

| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | S1      |
| GND             | P4      |
| TX2–            | S2      |
| TX2+            | S3      |
| CD2–            | R4      |
| CD2+            | P5      |
| RX2+            | R5      |
| RX2–            | S4      |
| V <sub>CC</sub> | S5      |
| GND             | S6      |
| RX1–            | P6      |
| RX1+            | R6      |
| CD1–            | S7      |
| CD1+            | R7      |
| TX1–            | P7      |
| TX1+            | P8      |
| V <sub>CC</sub> | R8      |
| GND             | S8      |
| V <sub>CC</sub> | S9      |
| GND             | R9      |
| CLKIN           | P9      |
| RA4             | S10     |
| RA3             | R10     |
| RA2             | S11     |
| RA1             | P10     |
| RA0             | R11     |
| V <sub>CC</sub> | S12     |
| GND             | R12     |
| MLOAD           | P11     |
| CDEC            | S13     |
| WR              | R13     |
| RD              | S14     |
| D7              | P12     |
| D6              | R14     |
| D5              | S15     |
| D4              | P13     |
| D3              | P14     |
| D2              | R15     |
| D1              | R16     |
| D0              | R16     |

| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | N13     |
| GND             | P15     |
| IRC             | N14     |
| IRE             | P16     |
| IRD             | N15     |
| COLN            | N16     |
| V <sub>CC</sub> | M15     |
| GND             | M14     |
| PKEN            | L14     |
| RXM             | L15     |
| BUFEN           | M16     |
| RDY             | L16     |
| ELI             | K16     |
| RTI             | K14     |
| STR1            | K15     |
| V <sub>CC</sub> | J16     |
| GND             | J15     |
| STR0            | J14     |
| ACTND           | H16     |
| ANYXND          | H15     |
| ACK0            | H14     |
| MRXC            | G14     |
| MEN             | G15     |
| MRXD            | G16     |
| MCRS            | F16     |
| V <sub>CC</sub> | F14     |
| GND             | F15     |
| ACKI            | E15     |
| ACTNS           | E14     |
| ANYXNS          | E16     |
| PCOMP           | D16     |
| RXI13–          | D15     |
| RXI13+          | D14     |
| TXO13P+         | C16     |
| TXO13–          | C15     |
| TXO13+          | B16     |
| TXO13P–         | B15     |
| V <sub>CC</sub> | D13     |
| GND             | C14     |

Note: NC = No Connect

2.0 Connection Diagram—160 Pin PGA Package (Continued)

|   |                 |                 |        |                 |                 |        |                 |                 |                 |                 |        |                 |                 |                 |                 |         |
|---|-----------------|-----------------|--------|-----------------|-----------------|--------|-----------------|-----------------|-----------------|-----------------|--------|-----------------|-----------------|-----------------|-----------------|---------|
| S | V <sub>CC</sub> | TX2-            | TX2+   | RX2-            | V <sub>CC</sub> | GND    | CD1-            | GND             | V <sub>CC</sub> | RA4             | RA2    | V <sub>CC</sub> | CDEC            | RD              | D5              | D1      |
|   | 98              | 96              | 95     | 91              | 90              | 89     | 86              | 81              | 80              | 77              | 75     | 72              | 69              | 67              | 64              | 60      |
| R | GND             | CD3-            | TX3-   | CD2-            | RX2+            | RX1+   | CD1+            | V <sub>CC</sub> | GND             | RA3             | RA0    | GND             | WR              | D6              | D2              | D0      |
|   | 105             | 101             | 99     | 94              | 92              | 87     | 85              | 82              | 79              | 76              | 73     | 71              | 68              | 65              | 61              | 59      |
| P | V <sub>CC</sub> | RX3+            | CD3+   | GND             | CD2+            | RX1-   | TX1-            | TX1+            | CLKIN           | RA1             | MLOAD  | D7              | D4              | D3              | GND             | IRE     |
|   | 106             | 104             | 102    | 97              | 93              | 88     | 84              | 83              | 78              | 74              | 70     | 66              | 63              | 62              | 57              | 55      |
| N | CD4+            | RX4+            | RX3-   | TX3+            |                 |        |                 |                 |                 |                 |        | V <sub>CC</sub> | IRC             | IRD             | COLN            |         |
|   | 109             | 108             | 103    | 100             |                 |        |                 |                 |                 |                 |        | 58              | 56              | 54              | 53              |         |
| M | TX4-            | CD4-            | RX4-   |                 |                 |        |                 |                 |                 |                 |        |                 | GND             | V <sub>CC</sub> | BUFEN           |         |
|   | 112             | 110             | 107    |                 |                 |        |                 |                 |                 |                 |        |                 | 51              | 52              | 48              |         |
| L | V <sub>CC</sub> | GND             | TX4+   |                 |                 |        |                 |                 |                 |                 |        |                 | PKEN            | RXM             | RDY             |         |
|   | 114             | 113             | 111    |                 |                 |        |                 |                 |                 |                 |        |                 | 50              | 49              | 47              |         |
| K | CD5-            | TX5-            | TX5+   |                 |                 |        |                 |                 |                 |                 |        |                 | RTI             | STR1            | ELI             |         |
|   | 117             | 115             | 116    |                 |                 |        |                 |                 |                 |                 |        |                 | 45              | 44              | 46              |         |
| J | CD5+            | RX5+            | GND    |                 |                 |        |                 |                 |                 |                 |        |                 | STR0            | GND             | V <sub>CC</sub> |         |
|   | 118             | 120             | 121    |                 |                 |        |                 |                 |                 |                 |        |                 | 41              | 42              | 43              |         |
| H | RX5-            | V <sub>CC</sub> | RX16+  |                 |                 |        |                 |                 |                 |                 |        |                 | ACK0            | ANYXND          | ACTND           |         |
|   | 119             | 122             | 125    |                 |                 |        |                 |                 |                 |                 |        |                 | 38              | 39              | 40              |         |
| G | NC              | RX16-           | TX06+  |                 |                 |        |                 |                 |                 |                 |        |                 | MRXC            | MEN             | MRXD            |         |
|   | 123             | 126             | 129    |                 |                 |        |                 |                 |                 |                 |        |                 | 37              | 36              | 35              |         |
| F | NC              | TX06-           | TX07P- |                 |                 |        |                 |                 |                 |                 |        |                 | V <sub>CC</sub> | GND             | MCRS            |         |
|   | 124             | 128             | 133    |                 |                 |        |                 |                 |                 |                 |        |                 | 33              | 32              | 34              |         |
| E | TX06P+          | TX06P-          | TX07+  |                 |                 |        |                 |                 |                 |                 |        |                 | ACTNS           | ACK1            | ANYXNS          |         |
|   | 127             | 130             | 134    |                 |                 |        |                 |                 |                 |                 |        |                 | 30              | 31              | 29              |         |
| D | GND             | TX07-           | GND    | TX08P+          |                 |        |                 |                 |                 |                 |        | V <sub>CC</sub> | RX113+          | RX113-          | PCOMP           |         |
|   | 131             | 135             | 139    | 143             |                 |        |                 |                 |                 |                 |        | 21              | 26              | 27              | 28              |         |
| C | V <sub>CC</sub> | RX17-           | TX08-  | TX08P-          | TX09-           | TX09P+ | V <sub>CC</sub> | TX010+          | TX010P-         | TX011+          | GND    | V <sub>CC</sub> | TX012P+         | GND             | TX013-          | TX013P+ |
|   | 132             | 138             | 144    | 146             | 151             | 152    | 156             | 2               | 3               | 7               | 12     | 13              | 16              | 20              | 24              | 25      |
| B | TX07P+          | RX18+           | RX18-  | GND             | TX09+           | RX19+  | RX110+          | RX110-          | GND             | V <sub>CC</sub> | RX111+ | RX111-          | RX112-          | TX012-          | TX013P-         | TX013+  |
|   | 136             | 141             | 142    | 147             | 150             | 153    | 157             | 158             | 4               | 5               | 10     | 11              | 15              | 17              | 22              | 23      |
| A | RX17+           | V <sub>CC</sub> | TX08+  | V <sub>CC</sub> | TX09P-          | RX19-  | GND             | TX010P+         | TX010-          | TX011P-         | TX011- | TX011P+         | RX112+          | TX012+          | TX012P-         |         |
|   | 137             | 140             | 145    | 148             | 149             | 154    | 155             | 159             | 1               | 6               | 8      | 9               | 14              | 18              | 19              |         |
|   | 1               | 2               | 3      | 4               | 5               | 6      | 7               | 8               | 9               | 0               | 1      | 2               | 3               | 4               | 5               | 6       |

**RIC  
DP83950**

TL/F/11096-3

**Bottom View**  
**1-5 AUI + 6-13 T.P. Ports**  
**Order Number DP83950BNU**  
**See NS Package Number UP159A**

## 2.0 Connection Diagram—160 Pin PGA Package (Continued)

Pin Table (1–7 AUI + 8–13 T.P. Ports)

| Pin Name        | Pin No. |
|-----------------|---------|
| TXO12P–         | A15     |
| TXO12+          | A14     |
| TXO12–          | B14     |
| TXO12P+         | C13     |
| RXI12–          | B13     |
| RXI12+          | A13     |
| V <sub>CC</sub> | C12     |
| GND             | C11     |
| RXI11–          | B12     |
| RXI11+          | B11     |
| TXO11P+         | A12     |
| TXO11–          | A11     |
| TXO11+          | C10     |
| TXO11P–         | A10     |
| V <sub>CC</sub> | B10     |
| GND             | B9      |
| TXO10P–         | C9      |
| TXO10+          | C8      |
| TXO10–          | A9      |
| TXO10P+         | A8      |
| RXI10–          | B8      |
| RXI10+          | B7      |
| V <sub>CC</sub> | C7      |
| GND             | A7      |
| RXI9–           | A6      |
| RXI9+           | B6      |
| TXO9P+          | C6      |
| TXO9–           | C5      |
| TXO9+           | B5      |
| TXO9P–          | A5      |
| V <sub>CC</sub> | A4      |
| GND             | B4      |
| TXO8P–          | C4      |
| TXO8+           | A3      |
| TXO8–           | C3      |
| TXO8P+          | D4      |
| RXI8–           | B3      |
| RXI8+           | B2      |
| V <sub>CC</sub> | A2      |
| GND             | D3      |

| Pin Name        | Pin No. |
|-----------------|---------|
| RX7+            | C2      |
| RX7–            | A1      |
| CD7+            | B1      |
| CD7–            | D2      |
| TX7+            | E3      |
| TX7–            | F3      |
| V <sub>CC</sub> | C1      |
| GND             | D1      |
| TX6–            | E2      |
| TX6+            | G3      |
| CD6–            | F2      |
| CD6+            | E1      |
| RX6+            | G2      |
| RX6–            | H3      |
| NC              | F1      |
| NC              | G1      |
| V <sub>CC</sub> | H2      |
| GND             | J3      |
| RX5+            | J2      |
| RX5–            | H1      |
| CD5+            | J1      |
| CD5–            | K1      |
| TX5+            | K3      |
| TX5–            | K2      |
| V <sub>CC</sub> | L1      |
| GND             | L2      |
| TX4–            | M1      |
| TX4+            | L3      |
| CD4–            | M2      |
| CD4+            | N1      |
| RX4+            | N2      |
| RX4–            | M3      |
| V <sub>CC</sub> | P1      |
| GND             | R1      |
| RX3+            | P2      |
| RX3–            | N3      |
| CD3+            | P3      |
| CD3–            | R2      |
| TX3+            | N4      |
| TX3–            | R3      |

| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | S1      |
| GND             | P4      |
| TX2–            | S2      |
| TX2+            | S3      |
| CD2–            | R4      |
| CD2+            | P5      |
| RX2+            | R5      |
| RX2–            | S4      |
| V <sub>CC</sub> | S5      |
| GND             | S6      |
| RX1–            | P6      |
| RX1+            | R6      |
| CD1–            | S7      |
| CD1+            | R7      |
| TX1–            | P7      |
| TX1+            | P8      |
| V <sub>CC</sub> | R8      |
| GND             | S8      |
| V <sub>CC</sub> | S9      |
| GND             | R9      |
| CLKIN           | P9      |
| RA4             | S10     |
| RA3             | R10     |
| RA2             | S11     |
| RA1             | P10     |
| RA0             | R11     |
| V <sub>CC</sub> | S12     |
| GND             | R12     |
| MLOAD           | P11     |
| CDEC            | S13     |
| WR              | R13     |
| RD              | S14     |
| D7              | P12     |
| D6              | R14     |
| D5              | S15     |
| D4              | P13     |
| D3              | P14     |
| D2              | R15     |
| D1              | S16     |
| D0              | R16     |

| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | N13     |
| GND             | P15     |
| IRC             | N14     |
| IRE             | P16     |
| IRD             | N15     |
| COLN            | N16     |
| V <sub>CC</sub> | M15     |
| GND             | M14     |
| PKEN            | L14     |
| RXM             | L15     |
| BUFEN           | M16     |
| RDY             | L16     |
| ELI             | K16     |
| RTI             | K14     |
| STR1            | K15     |
| V <sub>CC</sub> | J16     |
| GND             | J15     |
| STR0            | J14     |
| ACTND           | H16     |
| ANYXND          | H15     |
| ACKO            | H14     |
| MRXC            | G14     |
| MEN             | G15     |
| MRXD            | G16     |
| MCRS            | F16     |
| V <sub>CC</sub> | F14     |
| GND             | F15     |
| ACKI            | E15     |
| ACTNS           | E14     |
| ANYXNS          | E16     |
| PCOMP           | D16     |
| RXI13–          | D15     |
| RXI13+          | D14     |
| TXO13P+         | C16     |
| TXO13–          | C15     |
| TXO13+          | B16     |
| TXO13P–         | B15     |
| V <sub>CC</sub> | D13     |
| GND             | C14     |

Note: NC = No Connect



2.0 Connection Diagram—160 Pin PGA Package (Continued)

|   |                 |                 |       |                 |                 |        |                 |                 |                 |                 |        |                 |                 |                 |                 |         |
|---|-----------------|-----------------|-------|-----------------|-----------------|--------|-----------------|-----------------|-----------------|-----------------|--------|-----------------|-----------------|-----------------|-----------------|---------|
| S | V <sub>CC</sub> | TX2-            | TX2+  | RX2-            | V <sub>CC</sub> | GND    | CD1-            | GND             | V <sub>CC</sub> | RA4             | RA2    | V <sub>CC</sub> | CDEC            | RD              | D5              | D1      |
|   | 98              | 96              | 95    | 91              | 90              | 89     | 86              | 81              | 80              | 77              | 75     | 72              | 69              | 67              | 64              | 60      |
| R | GND             | CD3-            | TX3-  | CD2-            | RX2+            | RX1+   | CD1+            | V <sub>CC</sub> | GND             | RA3             | RA0    | GND             | WR              | D6              | D2              | D0      |
|   | 105             | 101             | 99    | 94              | 92              | 87     | 85              | 82              | 79              | 76              | 73     | 71              | 68              | 65              | 61              | 59      |
| P | V <sub>CC</sub> | RX3+            | CD3+  | GND             | CD2+            | RX1-   | TX1-            | TX1+            | CLKIN           | RA1             | MLOAD  | D7              | D4              | D3              | GND             | IRE     |
|   | 106             | 104             | 102   | 97              | 93              | 88     | 84              | 83              | 78              | 74              | 70     | 66              | 63              | 62              | 57              | 55      |
| N | CD4+            | RX4+            | RX3-  | TX3+            |                 |        |                 |                 |                 |                 |        | V <sub>CC</sub> | IRC             | IRD             | COLN            |         |
|   | 109             | 108             | 103   | 100             |                 |        |                 |                 |                 |                 |        | 58              | 56              | 54              | 53              |         |
| M | TX4-            | CD4-            | RX4-  |                 |                 |        |                 |                 |                 |                 |        |                 | GND             | V <sub>CC</sub> | BUFEN           |         |
|   | 112             | 110             | 107   |                 |                 |        |                 |                 |                 |                 |        |                 | 51              | 52              | 48              |         |
| L | V <sub>CC</sub> | GND             | TX4+  |                 |                 |        |                 |                 |                 |                 |        |                 | PKEN            | RXM             | RDY             |         |
|   | 114             | 113             | 111   |                 |                 |        |                 |                 |                 |                 |        |                 | 50              | 49              | 47              |         |
| K | CD5-            | TX5-            | TX5+  |                 |                 |        |                 |                 |                 |                 |        |                 | RTI             | STRT            | ELI             |         |
|   | 117             | 115             | 116   |                 |                 |        |                 |                 |                 |                 |        |                 | 45              | 44              | 46              |         |
| J | CD5+            | RX5+            | GND   |                 |                 |        |                 |                 |                 |                 |        |                 | STR0            | GND             | V <sub>CC</sub> |         |
|   | 118             | 120             | 121   |                 |                 |        |                 |                 |                 |                 |        |                 | 41              | 42              | 43              |         |
| H | RX5-            | V <sub>CC</sub> | RX6-  |                 |                 |        |                 |                 |                 |                 |        |                 | ACK0            | ANYXND          | ACTND           |         |
|   | 119             | 122             | 125   |                 |                 |        |                 |                 |                 |                 |        |                 | 38              | 39              | 40              |         |
| G | NC              | RX6+            | TX6+  |                 |                 |        |                 |                 |                 |                 |        |                 | MRXC            | MEN             | MRXD            |         |
|   | 123             | 126             | 129   |                 |                 |        |                 |                 |                 |                 |        |                 | 37              | 36              | 35              |         |
| F | NC              | CD6-            | TX7-  |                 |                 |        |                 |                 |                 |                 |        |                 | V <sub>CC</sub> | GND             | MCRS            |         |
|   | 124             | 128             | 133   |                 |                 |        |                 |                 |                 |                 |        |                 | 33              | 32              | 34              |         |
| E | CD6+            | TX6-            | TX7+  |                 |                 |        |                 |                 |                 |                 |        |                 | ACTNS           | ACKI            | ANYXNS          |         |
|   | 127             | 130             | 134   |                 |                 |        |                 |                 |                 |                 |        |                 | 30              | 31              | 29              |         |
| D | GND             | CD7-            | GND   | TX08P+          |                 |        |                 |                 |                 |                 |        | V <sub>CC</sub> | RXI13+          | RXI13-          | PCOMP           |         |
|   | 131             | 135             | 139   | 143             |                 |        |                 |                 |                 |                 |        | 21              | 26              | 27              | 28              |         |
| C | V <sub>CC</sub> | RX7+            | TX08- | TX08P-          | TX09-           | TX09P+ | V <sub>CC</sub> | TX010+          | TX010P-         | TX011+          | GND    | V <sub>CC</sub> | TX012P+         | GND             | TX013-          | TX013P+ |
|   | 132             | 138             | 144   | 146             | 151             | 152    | 156             | 2               | 3               | 7               | 12     | 13              | 16              | 20              | 24              | 25      |
| B | CD7+            | RX18+           | RX18- | GND             | TX09+           | RX19+  | RX110+          | RX110-          | GND             | V <sub>CC</sub> | RX111+ | RX111-          | RX112-          | TX012-          | TX013P-         | TX013+  |
|   | 136             | 141             | 142   | 147             | 150             | 153    | 157             | 158             | 4               | 5               | 10     | 11              | 15              | 17              | 22              | 23      |
| A | RX7-            | V <sub>CC</sub> | TX08+ | V <sub>CC</sub> | TX09P-          | RX19-  | GND             | TX010P+         | TX010-          | TX011P-         | TX011- | TX011P+         | RX112+          | TX012+          | TX012P-         |         |
|   | 137             | 140             | 145   | 148             | 149             | 154    | 155             | 159             | 1               | 6               | 8      | 9               | 14              | 18              | 19              |         |
|   | 1               | 2               | 3     | 4               | 5               | 6      | 7               | 8               | 9               | 0               | 1      | 2               | 3               | 4               | 5               | 6       |

RIC  
DP83950

TL/F/11096-4

Bottom View  
1-7 AUI + 8-13 T.P. Ports  
Order Number DP83950BNU  
See NS Package Number UP159A

## 2.0 Connection Diagram—160 Pin PGA Package (Continued)

Pin Table (All AUI Ports)

| Pin Name        | Pin No. |
|-----------------|---------|
| TX12-           | A15     |
| TX12+           | A14     |
| CD12-           | B14     |
| CD12+           | C13     |
| RX12+           | B13     |
| RX12-           | A13     |
| V <sub>CC</sub> | C12     |
| GND             | C11     |
| RX11+           | B12     |
| RX11-           | B11     |
| CD11+           | A12     |
| CD11-           | A11     |
| TX11+           | C10     |
| TX11-           | A10     |
| V <sub>CC</sub> | B10     |
| GND             | B9      |
| TX10-           | C9      |
| TX10+           | C8      |
| CD10-           | A9      |
| CD10+           | A8      |
| RX10+           | B8      |
| RX10-           | B7      |
| V <sub>CC</sub> | C7      |
| GND             | A7      |
| RX9+            | A6      |
| RX9-            | B6      |
| CD9+            | C6      |
| CD9-            | C5      |
| TX9+            | B5      |
| TX9-            | A5      |
| V <sub>CC</sub> | A4      |
| GND             | B4      |
| TX8-            | C4      |
| TX8+            | A3      |
| CD8-            | C3      |
| CD8+            | D4      |
| RX8+            | B3      |
| RX8-            | B2      |
| V <sub>CC</sub> | A2      |
| GND             | D3      |

| Pin Name        | Pin No. |
|-----------------|---------|
| RX7+            | C2      |
| RX7-            | A1      |
| CD7+            | B1      |
| CD7-            | D2      |
| TX7+            | E3      |
| TX7-            | F3      |
| V <sub>CC</sub> | C1      |
| GND             | D1      |
| TX6-            | E2      |
| TX6+            | G3      |
| CD6-            | F2      |
| CD6+            | E1      |
| RX6+            | G2      |
| RX6-            | H3      |
| NC              | F1      |
| NC              | G1      |
| V <sub>CC</sub> | H2      |
| GND             | J3      |
| RX5+            | J2      |
| RX5-            | H1      |
| CD5+            | J1      |
| CD5-            | K1      |
| TX5+            | K3      |
| TX5-            | K2      |
| V <sub>CC</sub> | L1      |
| GND             | L2      |
| TX4-            | M1      |
| TX4+            | L3      |
| CD4-            | M2      |
| CD4+            | N1      |
| RX4+            | N2      |
| RX4-            | M3      |
| V <sub>CC</sub> | P1      |
| GND             | R1      |
| RX3+            | P2      |
| RX3-            | N3      |
| CD3+            | P3      |
| CD3-            | R2      |
| TX3+            | N4      |
| TX3-            | R3      |

| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | S1      |
| GND             | P4      |
| TX2-            | S2      |
| TX2+            | S3      |
| CD2-            | R4      |
| CD2+            | P5      |
| RX2+            | R5      |
| RX2-            | S4      |
| V <sub>CC</sub> | S5      |
| GND             | S6      |
| RX1-            | P6      |
| RX1+            | R6      |
| CD1-            | S7      |
| CD1+            | R7      |
| TX1-            | P7      |
| TX1+            | P8      |
| V <sub>CC</sub> | R8      |
| GND             | S8      |
| V <sub>CC</sub> | S9      |
| GND             | R9      |
| CLKIN           | P9      |
| RA4             | S10     |
| RA3             | R10     |
| RA2             | S11     |
| RA1             | P10     |
| RA0             | R11     |
| V <sub>CC</sub> | S12     |
| GND             | R12     |
| MLOAD           | P11     |
| CDEC            | S13     |
| WR              | R13     |
| RD              | S14     |
| D7              | P12     |
| D6              | R14     |
| D5              | S15     |
| D4              | P13     |
| D3              | P14     |
| D2              | R15     |
| D1              | S16     |
| D0              | R16     |

| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | N13     |
| GND             | P15     |
| IRC             | N14     |
| IRE             | P16     |
| IRD             | N15     |
| COLN            | N16     |
| V <sub>CC</sub> | M15     |
| GND             | M14     |
| PKEN            | L14     |
| RXM             | L15     |
| BUFEN           | M16     |
| RDY             | L16     |
| ELI             | K16     |
| RTI             | K14     |
| STR1            | K15     |
| V <sub>CC</sub> | J16     |
| GND             | J15     |
| STR0            | J14     |
| ACTND           | H16     |
| ANYXND          | H15     |
| ACK0            | H14     |
| MRXC            | G14     |
| MEN             | G15     |
| MRXD            | G16     |
| MCRS            | F16     |
| V <sub>CC</sub> | F14     |
| GND             | F15     |
| ACKI            | E15     |
| ACTNS           | E14     |
| ANYXNS          | E16     |
| PCOMP           | D16     |
| RX13+           | D15     |
| RX13-           | D14     |
| CD13+           | C16     |
| CD13-           | C15     |
| TX13+           | B16     |
| TX13-           | B15     |
| V <sub>CC</sub> | D13     |
| GND             | C14     |

Note: NC = No Connect

2.0 Connection Diagram—160 Pin PGA Package (Continued)

|   |                 |                 |      |                 |                 |      |                 |                 |                 |                 |       |                 |                 |                 |                 |       |
|---|-----------------|-----------------|------|-----------------|-----------------|------|-----------------|-----------------|-----------------|-----------------|-------|-----------------|-----------------|-----------------|-----------------|-------|
| S | V <sub>CC</sub> | TX2-            | TX2+ | RX2-            | V <sub>CC</sub> | GND  | CD1-            | GND             | V <sub>CC</sub> | RA4             | RA2   | V <sub>CC</sub> | CDEC            | R <sub>D</sub>  | D5              | D1    |
|   | 98              | 96              | 95   | 91              | 90              | 89   | 86              | 81              | 80              | 77              | 75    | 72              | 69              | 67              | 64              | 60    |
| R | GND             | CD3-            | TX3- | CD2-            | RX2+            | RX1+ | CD1+            | V <sub>CC</sub> | GND             | RA3             | RA0   | GND             | WR              | D6              | D2              | 00    |
|   | 105             | 101             | 99   | 94              | 92              | 87   | 85              | 82              | 79              | 76              | 73    | 71              | 68              | 65              | 61              | 59    |
| P | V <sub>CC</sub> | RX3+            | CD3+ | GND             | CD2+            | RX1- | TX1-            | TX1+            | CLKIN           | RA1             | MLOAD | D7              | D4              | D3              | GND             | IRE   |
|   | 106             | 104             | 102  | 97              | 93              | 88   | 84              | 83              | 78              | 74              | 70    | 66              | 63              | 62              | 57              | 55    |
| N | CD4+            | RX4+            | RX3- | TX3+            |                 |      |                 |                 |                 |                 |       | V <sub>CC</sub> | IRC             | IRD             | COLN            |       |
|   | 109             | 108             | 103  | 100             |                 |      |                 |                 |                 |                 |       | 58              | 56              | 54              | 53              |       |
| M | TX4-            | CD4-            | RX4- |                 |                 |      |                 |                 |                 |                 |       |                 | GND             | V <sub>CC</sub> | BUFEN           |       |
|   | 112             | 110             | 107  |                 |                 |      |                 |                 |                 |                 |       |                 | 51              | 52              | 48              |       |
| L | V <sub>CC</sub> | GND             | TX4+ |                 |                 |      |                 |                 |                 |                 |       |                 | PKEN            | RXM             | RDY             |       |
|   | 114             | 113             | 111  |                 |                 |      |                 |                 |                 |                 |       |                 | 50              | 49              | 47              |       |
| K | CD5-            | TX5-            | TX5+ |                 |                 |      |                 |                 |                 |                 |       |                 | RTI             | STR1            | ELI             |       |
|   | 117             | 115             | 116  |                 |                 |      |                 |                 |                 |                 |       |                 | 45              | 44              | 46              |       |
| J | CD5+            | RX5+            | GND  |                 |                 |      |                 |                 |                 |                 |       |                 | STR0            | GND             | V <sub>CC</sub> |       |
|   | 118             | 120             | 121  |                 |                 |      |                 |                 |                 |                 |       |                 | 41              | 42              | 43              |       |
| H | RX5-            | V <sub>CC</sub> | RX6- |                 |                 |      |                 |                 |                 |                 |       |                 | ACK0            | ANYXND          | ACTND           |       |
|   | 119             | 122             | 125  |                 |                 |      |                 |                 |                 |                 |       |                 | 38              | 39              | 40              |       |
| G | NC              | RX6+            | TX6+ |                 |                 |      |                 |                 |                 |                 |       |                 | MRXC            | MEN             | MRXD            |       |
|   | 123             | 126             | 129  |                 |                 |      |                 |                 |                 |                 |       |                 | 37              | 36              | 35              |       |
| F | NC              | CD6-            | TX7- |                 |                 |      |                 |                 |                 |                 |       |                 | V <sub>CC</sub> | GND             | MCRS            |       |
|   | 124             | 128             | 133  |                 |                 |      |                 |                 |                 |                 |       |                 | 33              | 32              | 34              |       |
| E | CD6+            | TX6-            | TX7+ |                 |                 |      |                 |                 |                 |                 |       |                 | ACTNS           | ACK1            | ANYXNS          |       |
|   | 127             | 130             | 134  |                 |                 |      |                 |                 |                 |                 |       |                 | 30              | 31              | 29              |       |
| D | GND             | CD7-            | GND  | CD8+            |                 |      |                 |                 |                 |                 |       | V <sub>CC</sub> | RX13-           | RX13+           | PCOMP           |       |
|   | 131             | 135             | 139  | 143             |                 |      |                 |                 |                 |                 |       | 21              | 26              | 27              | 28              |       |
| C | V <sub>CC</sub> | RX7+            | CD8- | TX8-            | CD9-            | CD9+ | V <sub>CC</sub> | TX10+           | TX10-           | TX11+           | GND   | V <sub>CC</sub> | CD12+           | GND             | CD13-           | CD13+ |
|   | 132             | 138             | 144  | 146             | 151             | 152  | 156             | 2               | 3               | 7               | 12    | 13              | 16              | 20              | 24              | 25    |
| B | CD7+            | RX8-            | RX8+ | GND             | TX9+            | RX9- | RX10-           | RX10+           | GND             | V <sub>CC</sub> | RX11- | RX11+           | RX12+           | CD12-           | TX13-           | TX13+ |
|   | 136             | 141             | 142  | 147             | 150             | 153  | 157             | 158             | 4               | 5               | 10    | 11              | 15              | 17              | 22              | 23    |
| A | RX7-            | V <sub>CC</sub> | TX8+ | V <sub>CC</sub> | TX9-            | RX9+ | GND             | CD10+           | CD10-           | TX11-           | CD11- | CD11+           | RX12-           | TX12+           | TX12-           |       |
|   | 137             | 140             | 145  | 148             | 149             | 154  | 155             | 159             | 1               | 6               | 8     | 9               | 14              | 18              | 19              |       |
|   | 1               | 2               | 3    | 4               | 5               | 6    | 7               | 8               | 9               | 0               | 1     | 2               | 3               | 4               | 5               | 6     |

RIC  
DP83950

Bottom View  
All AUI Ports

Order Number DP83950BNU  
See NS Package Number UP159A

TL/F/11096-5

### 3.0 Pin Descriptions

| Pin No.                                                                                           | Pin Name          | Driver Type | I/O | Description                                        |
|---------------------------------------------------------------------------------------------------|-------------------|-------------|-----|----------------------------------------------------|
| <b>NETWORK INTERFACE PINS (On-Chip Transceiver Mode)</b>                                          |                   |             |     |                                                    |
|                                                                                                   | RX12- to RX113-   | TP          | I   | Twisted Pair Receive Input Negative                |
|                                                                                                   | RX12+ to RX113+   | TP          | I   | Twisted Pair Receive Input Positive                |
|                                                                                                   | TXOP2- to TXOP13- | TT          | O   | Twisted Pair Pre-emphasis Transmit Output Negative |
|                                                                                                   | TXO2- to TXO13-   | TT          | O   | Twisted Pair Transmit Output Negative              |
|                                                                                                   | TXO2+ to TXO13+   | TT          | O   | Twisted Pair Transmit Output Positive              |
|                                                                                                   | TXOP2+ to TXOP13+ | TT          | O   | Twisted Pair Pre-emphasis Transmit Output Positive |
|                                                                                                   | CD1+              | AL          | I   | AUI Collision Detect Input Positive                |
|                                                                                                   | CD1-              | AL          | I   | AUI Collision Detect Input Negative                |
|                                                                                                   | RX1+              | AL          | I   | AUI Receive Input Positive                         |
|                                                                                                   | RX1-              | AL          | I   | AUI Receive Input Negative                         |
|                                                                                                   | TX1+              | AD          | O   | AUI Transmit Output Positive                       |
|                                                                                                   | TX1-              | AD          | O   | AUI Transmit Output Negative                       |
| <b>NETWORK INTERFACE PINS (External Transceiver Mode AUI Signal Level Compatibility Selected)</b> |                   |             |     |                                                    |
|                                                                                                   | TX2+ to TX13+     | AL          | O   | Transmit Output Positive                           |
|                                                                                                   | TX2- to TX13-     | AL          | O   | Transmit Output Negative                           |
|                                                                                                   | CD2+ to CD13+     | AL          | I   | Collision Input Positive                           |
|                                                                                                   | CD2- to CD13-     | AL          | I   | Collision Input Negative                           |
|                                                                                                   | RX2+ to RX13+     | AL          | I   | Receive Input Positive                             |
|                                                                                                   | RX2- to RX13-     | AL          | I   | Receive Input Negative                             |
|                                                                                                   | CD1+              | AL          | I   | AUI Collision Detect Input Positive                |
|                                                                                                   | CD1-              | AL          | I   | AUI Collision Detect Input Negative                |
|                                                                                                   | RX1+              | AL          | I   | AUI Receive Input Positive                         |
|                                                                                                   | RX1-              | AL          | I   | AUI Receive Input Negative                         |
|                                                                                                   | TX1+              | AD          | O   | AUI Transmit Output Positive                       |
|                                                                                                   | TX1-              | AD          | O   | AUI Transmit Output Negative                       |

**Note:** AD = AUI level and Drive compatible, TP = Twisted Pair interface compatible, AL = AUI Level compatible, TT = TTL compatible, I = Input, O = Output.

### 3.0 Pin Descriptions (Continued)

| Pin No.                   | Pin Name           | Driver Type | I/O  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------|--------------------|-------------|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PROCESSOR BUS PINS</b> |                    |             |      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|                           | RA0-RA4            | TT          | I    | <b>REGISTER ADDRESS INPUTS:</b> These five pins are used to select a register to be read or written. The state of these inputs are ignored when the read, write and mode load input strobes are high. (Even under these conditions these inputs must not be allowed to float at an undefined logic state).                                                                                                                                                                                                       |
|                           | STR0               | C           | O    | <b>DISPLAY UPDATE STROBE 0</b><br><b>Maximum Display Mode:</b> This signal controls the latching of display data for network ports 1 to 7 into the off chip display latches.<br><b>Minimum Display Mode:</b> This signal controls the latching of display data for the RIC into the off chip display latch.<br>During processor access cycles (read or write is asserted) this signal is inactive (high).                                                                                                        |
|                           | STR1               | C           | O    | <b>DISPLAY UPDATE STROBE 1</b><br><b>Maximum Display Mode:</b> This signal controls the latching of display data for network ports 8 to 13 into the off chip display latches.<br><b>Minimum Display Mode:</b> No operation<br>During processor access cycles (read or write is asserted) this signal is inactive (high).                                                                                                                                                                                         |
|                           | D0-D7              | TT          | B, Z | <b>DATA BUS</b><br><b>Display Update Cycles:</b> These pins become outputs providing display data and port address information. Address information only available in Maximum Display mode.<br><b>Processor Access Cycles:</b> Data input or output is performed via these pins. The read, write and mode load inputs control the direction of the signals.<br><b>Note:</b> The data pins remain in their display update function, i.e., asserted as outputs unless either the read or write strobe is asserted. |
|                           | BUFEN              | C           | O    | <b>BUFFER ENABLE:</b> This output controls the TRI-STATE® operation of the bus transceiver which provides the interface between the RIC's data pins and the processor's data bus.<br><b>Note:</b> The buffer enable output indicates the function of the data pins. When it is high they are performing display update cycles, when it is low a processor access or mode load cycle is occurring.                                                                                                                |
|                           | RDY                | C           | O    | <b>DATA READY STROBE:</b> The falling edge of this signal during a read cycle indicates that data is stable and valid for sampling. In write cycles the falling edge of $\overline{RDY}$ denotes that the write data has been latched by the RIC. Therefore data must have been available and stable for this operation to be successful.                                                                                                                                                                        |
|                           | $\overline{ELI}$   | C           | O    | <b>EVENT LOGGING INTERRUPT:</b> A low level on the $\overline{ELI}$ output indicates the RIC's hub management logic requires CPU attention. The interrupt is cleared by accessing the Port Event Recording register or Event Counter that produced it. All interrupt sources may be masked.                                                                                                                                                                                                                      |
|                           | $\overline{RTI}$   | C           | O    | <b>REAL TIME INTERRUPT:</b> A low level on the $\overline{RTI}$ output indicates the RIC's real time (packet specific) interrupt logic requires CPU attention. The interrupt is cleared by reading the Real Time Interrupt Status register. All interrupt sources may be masked.                                                                                                                                                                                                                                 |
|                           | $\overline{CDEC}$  | TT          | I    | <b>COUNTER DECREMENT:</b> A low level on the $\overline{CDEC}$ input strobe decrements all of the RIC's Port Event Counters by one. This input is internally synchronized and if necessary the operation of the signal is delayed if there is a simultaneous internally generated counting operation.                                                                                                                                                                                                            |
|                           | WR                 | TT          | I    | <b>WRITE STROBE:</b> Strobe from the CPU used to write an internal register defined by the RA0-RA4 inputs.                                                                                                                                                                                                                                                                                                                                                                                                       |
|                           | RD                 | TT          | I    | <b>READ STROBE:</b> Strobe from the CPU used to read an internal register defined by the RA0-RA4 inputs.                                                                                                                                                                                                                                                                                                                                                                                                         |
|                           | $\overline{MLOAD}$ | TT          | I    | <b>DEVICE RESET AND MODE LOAD:</b> When this input is low all of the RIC's state machines, counters and network ports are reset and held inactive. On the rising edge of $\overline{MLOAD}$ the logic levels present on the D0-7 pins and RA0-RA4 inputs are latched into the RIC's configuration registers. The rising edge of $\overline{MLOAD}$ also signals the beginning of the display test operation.                                                                                                     |

### 3.0 Pin Descriptions (Continued)

| Pin No.                   | Pin Name                 | Driver Type | I/O  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------------|--------------------------|-------------|------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>INTER-RIC BUS PINS</b> |                          |             |      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|                           | $\overline{\text{ACKI}}$ | TT          | I    | <b>ACKNOWLEDGE INPUT:</b> Input to the network ports' arbitration chain.                                                                                                                                                                                                                                                                                                                                                                                           |
|                           | $\overline{\text{ACKO}}$ | TT          | O    | <b>ACKNOWLEDGE OUTPUT:</b> Output from the network ports' arbitration chain.                                                                                                                                                                                                                                                                                                                                                                                       |
|                           | IRD                      | TT          | B, Z | <b>INTER-RIC DATA:</b> When asserted as an output this signal provides a serial data stream in NRZ format. The signal is asserted by a RIC when it is receiving data from one of its network segments. The default condition of this signal is to be an input. In this state it may be driven by other devices on the Inter-RIC bus.                                                                                                                               |
|                           | IRE                      | TT          | B, Z | <b>INTER-RIC ENABLE:</b> When asserted as an output this signal provides an activity framing enable for the serial data stream. The signal is asserted by a RIC when it is receiving data from one of its network segments. The default condition of this signal is to be an input. In this state it may be driven by other devices on the Inter-RIC bus.                                                                                                          |
|                           | IRC                      | TT          | B, Z | <b>INTER-RIC CLOCK:</b> When asserted as an output this signal provides a clock signal for the serial data stream. Data (IRD) is changed on the falling edge of the clock. The signal is asserted by a RIC when it is receiving data from one of its network segments. The default condition of this signal is to be an input. When an input IRD is sampled on the rising edge of the clock. In this state it may be driven by other devices on the Inter-RIC bus. |
|                           | COLN                     | TT          | B, Z | <b>COLLISION ON PORT N:</b> This denotes that a collision is occurring on the port receiving the data packet. The default condition of this signal is to be an input. In this state it may be driven by other devices on the Inter-RIC bus.                                                                                                                                                                                                                        |
|                           | PKEN                     | C           | O    | <b>PACKET ENABLE:</b> This output acts as an active high enable for an external bus transceiver (if required) for the IRE, IRC, IRD and COLN signals. When high the bus transceiver should be transmitting on to the bus, i.e., this RIC is driving the IRD, IRE, IRC and COLN bus lines. When low the bus transceiver should receive from the bus.                                                                                                                |
|                           | CLKIN                    | TT          | I    | <b>40 MHz CLOCK INPUT:</b> This input is used to generate the RIC's timing reference for the state machines, and phase lock loop decoder.                                                                                                                                                                                                                                                                                                                          |
|                           | ACTND                    | OD          | O    | <b>ACTIVITY ON PORT N DRIVE:</b> This output is active when the RIC is receiving data or collision information from one of its network segments.                                                                                                                                                                                                                                                                                                                   |
|                           | ACTNS                    | TT          | I    | <b>ACTIVITY ON PORT N SENSE:</b> This input senses when this or another RIC in a multi-RIC system is receiving data or collision information.                                                                                                                                                                                                                                                                                                                      |
|                           | ANYXND                   | OD          | O    | <b>ACTIVITY ON ANY PORT EXCLUDING PORT N DRIVE:</b> This output is active when a RIC is experiencing a transmit collision or multiple ports have active collisions on their network segments.                                                                                                                                                                                                                                                                      |
|                           | ANYXNS                   | TT          | I    | <b>ACTIVITY ON ANY PORT EXCLUDING PORT N SENSE:</b> This input senses when this RIC or other RICs in a multi-RIC system are experiencing transmit collisions or multiple ports have active collisions on their network segments.                                                                                                                                                                                                                                   |

### 3.0 Pin Descriptions (Continued)

| Pin No.                      | Pin Name | Driver Type | I/O  | Description                                                                                                                                                                                                                                                                                               |
|------------------------------|----------|-------------|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>MANAGEMENT BUS PINS</b>   |          |             |      |                                                                                                                                                                                                                                                                                                           |
|                              | MRXC     | TT          | O, Z | <b>MANAGEMENT RECEIVE CLOCK:</b> When asserted this signal provides a clock signal for the MRXD serial data stream. The MRXD signal is changed on the falling edge of this clock. The signal is asserted when a RIC is receiving data from one of its network segments. Otherwise the signal is inactive. |
|                              | MCRS     | TT          | B, Z | <b>MANAGEMENT CARRIER SENSE:</b> When asserted this signal provides an activity framing enable for the serial output data stream (MRXD). The signal is asserted when a RIC is receiving data from one of its network segments. Otherwise the signal is an input.                                          |
|                              | MRXD     | TT          | O, Z | <b>MANAGEMENT RECEIVE DATA:</b> When asserted this signal provides a serial data stream in NRZ format. The data stream is made up of the data packet and RIC status information. The signal is asserted when a RIC is receiving data from one of its network segments. Otherwise the signal is inactive.  |
|                              | MEN      | C           | O    | <b>MANAGEMENT BUS OUTPUT ENABLE:</b> This output acts as an active high enable for an external bus transceiver (if required) for the MRXC, MCRS and MRXD signals. When high the bus transceiver should be transmitting on to the bus.                                                                     |
|                              | PCOMP    | TT          | I    | <b>PACKET COMPRESS:</b> This input is used to activate the RIC's packet compress logic. A low level on this signal when MCRS is active will cause that packet to be compressed. If PCOMP is tied low all packets are compressed, if PCOMP is tied high packet compression is inhibited.                   |
| <b>POWER AND GROUND PINS</b> |          |             |      |                                                                                                                                                                                                                                                                                                           |
|                              | VCC      |             |      | Positive Supply                                                                                                                                                                                                                                                                                           |
|                              | GND      |             |      | Negative Supply                                                                                                                                                                                                                                                                                           |
| <b>EXTERNAL DECODER PINS</b> |          |             |      |                                                                                                                                                                                                                                                                                                           |
|                              | RXM      | TT          | O    | <b>RECEIVE DATA MANCHESTER FORMAT:</b> This output makes the data, in Manchester format, received by port N available for test purposes. If not used for testing this pin should be left open.                                                                                                            |

**Note:** TT = TTL compatible, B = Bi-directional, C = CMOS compatible, OD = Open Drain, I = Input, O = Output, Z = TRI-STATE

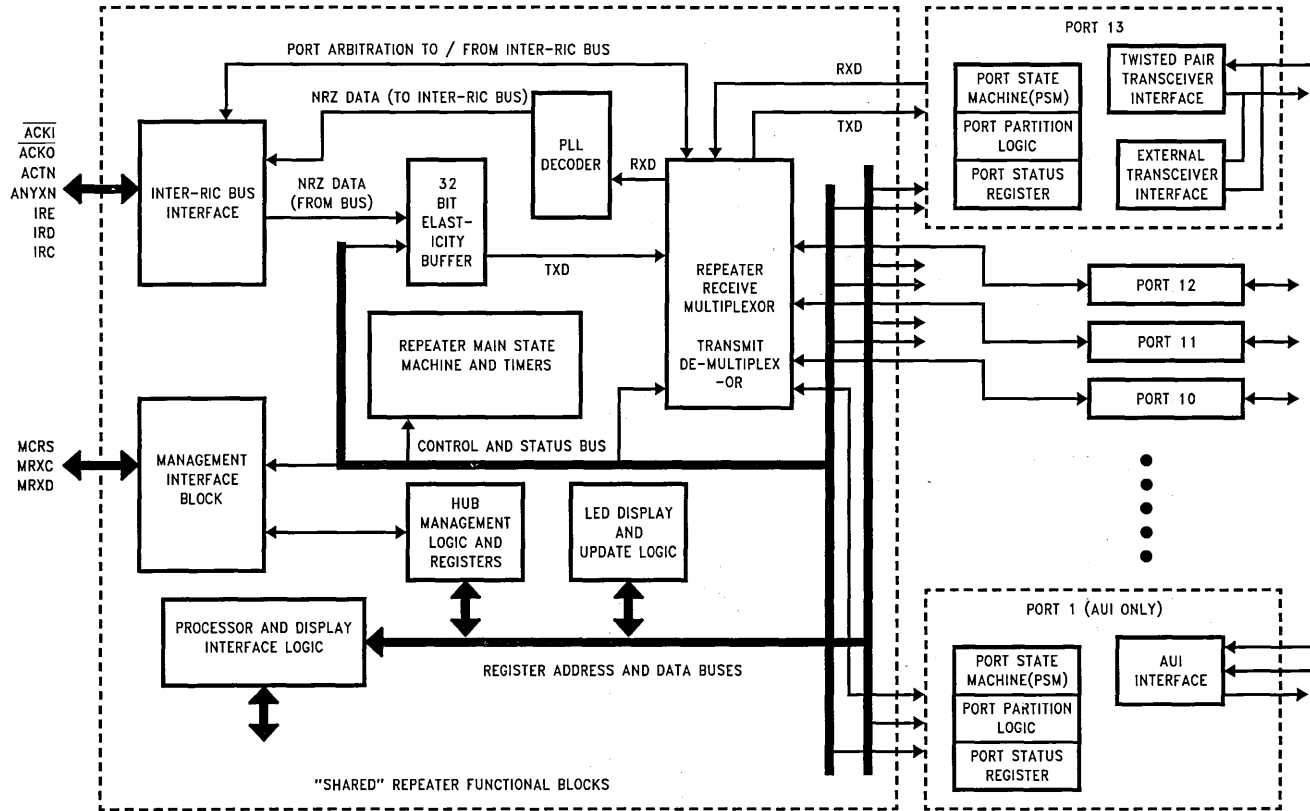


FIGURE 5.1

TL/F/11096-6



## 5.0 Functional Description

The I.E.E.E. repeater specification details a number of functions a repeater system must perform. These requirements allied with a need for the implementation to be multiport strongly favors the choice of a modular design style. In such a design, functionality is split between those tasks common to all data channels and those exclusive to each individual channel. The RIC follows this approach, certain functional blocks are replicated for each network attachment, (also known as a repeater port), and others are shared. The following section briefly describes the functional blocks in the RIC.

### 5.1 OVERVIEW OF RIC FUNCTIONS

#### Segment Specific Block: Network Port

As shown in the Block Diagram, the segment specific blocks consist of:

1. One or more physical layer interfaces.
2. A logic block required for performing repeater operations upon that particular segment. This is known as the "port" logic since it is the access "port" the segment has to the rest of the network.

This function is repeated 13 times in the RIC (one for each port) and is shown on the right side of the Block Diagram, *Figure 5.1*.

The physical layer interfaces provided depends upon the port under examination. Port 1 has an AUI compliant interface for use with AUI compatible transceiver boxes and cable. Ports 2 to 13 may be configured for use with one of two interfaces: twisted pair or an external transceiver. The former utilizes the RIC's on-chip 10BASE-T transceivers, the latter allows connection to external transceivers. When using the external transceiver mode the interface is AUI compatible. Although AUI compatible transceivers are supported the interface is not designed for use with an interface cable, thus the transceivers are necessarily internal to the repeater equipment.

Inside the port logic there are 3 distinct functions:

1. The port state machine "PSM" is required to perform data and collision repetition as described by the repeater specification, for example, it determines whether this port should be receiving from or transmitting to its network segment.
2. The port partition logic implements the segment partitioning algorithm. This algorithm is defined by the IEEE specification and is used to protect the network from malfunctioning segments.
3. The port status register reflects the current status of the port. It may be accessed by a system processor to obtain this status or to perform certain port configuration operations, such as port disable.

#### Shared Functional Blocks: Repeater Core Logic

The shared functional blocks consist of the Repeater Main State Machine (MSM) and Timers, a 32 bit Elasticity Buffer, PLL Decoder, and Receive and Transmit Multiplexors. These blocks perform the majority of the operations needed to fulfill the requirements of the IEEE repeater specification. When a packet is received by a port it is sent via the Receive Multiplexor to the PLL Decoder. Notification of the

data and collision status is sent to the main state machine via the receive multiplexor and collision activity status signals. This enables the main state machine to determine the source of the data to be repeated and the type of data to be transmitted. The transmit data may be either the received packet's data field or a preamble/jam pattern consisting of a 1010 . . . bit pattern.

Associated with the main state machine are a series of timers. These ensure various IEEE specification times (referred to as the TW1 to TW6 times) are fulfilled.

A repeater unit is required to meet the same signal jitter performance as any receiving node attached to a network segment. Consequently, a phase locked loop Manchester decoder is required so that the packet may be decoded, and the jitter accumulated over the receiving segment recovered. The decode logic outputs data in NRZ format with an associated clock and enable. In this form the packet is in a convenient format for transfer to other devices, such as network controllers and other RICs, via the Inter-RIC bus (described later). The data may then be re-encoded into Manchester data and transmitted.

Reception and transmission via physical layer transceiver units causes a loss of bits in the preamble field of a data packet. The repeater specification requires this loss to be compensated for. To accomplish this an elasticity buffer is employed to temporarily store bits in the data field of the packet.

The sequence of operation is as follows:

Soon after the network segment receiving the data packet has been identified, the RIC begins to transmit the packet preamble pattern (1010 . . . ) onto the other network segments. While the preamble is being transmitted the Elasticity Buffer monitors the decoded received clock and data signals (this is done via the Inter-RIC bus as described later). When the start of frame delimiter "SFD" is detected the received data stream is written into the elasticity buffer. Removal of data from the buffer for retransmission is not allowed until a valid length preamble pattern has been transmitted.

#### Inter-RIC Bus Interface

Using the RIC in a repeater system allows the design to be constructed with many more network attachments than can be supported by a single chip. The split of functions already described allows data packets and collision status to be transferred between multiple RICs, and at the same time the multiple RICs still behave as a single logical repeater. Since all RICs in a repeater system are identical and capable of performing any of the repetition operations, the failure of one RIC will not cause the failure of the entire system. This is an important issue in large multiport repeaters.

RICs communicate via a specialized interface known as the Inter-RIC bus. This allows the data packet to be transferred from the receiving RIC to the other RICs in the system. These RICs then transmit the data stream to their segments. Just as important as data transfer is the notification of collisions occurring across the network. The Inter-RIC bus has a set of status lines capable of conveying collision information between RICs to ensure their main state machines operate in the appropriate manner.

## 5.0 Functional Description (Continued)

### LED Interface and Hub Management Function

Repeater systems usually possess optical displays indicating network activity and the status of specific repeater operations. The RIC's display update block provides the system designer with a wide variety of indicators. The display updates are completely autonomous and merely require SSI logic devices to drive the display devices, usually made up of light emitting diodes, LEDs. The status display is very flexible allowing the user to choose those indicators appropriate for the specification of the equipment.

The RIC has been designed with special awareness for system designers implementing large repeaters possessing hub management capabilities. Hub management uses the unique position of repeaters in a network to gather statistics about the network segments they are attached to. The RIC provides hub management statistical data in 3 steps. Important events are gathered by the management block from logic blocks throughout the chip. These events may then be stored in on-chip latches or counted in on-chip counters according to user supplied latching and counting masks.

The fundamental task of a hub management system implementation is to associate the current packet and any management status information with the network segment, i.e., repeater port where the packet was received. The ideal system would place this combined data packet and status field in system memory for examination by hub management software. The ultimate function of the RIC's hub management support logic is to provide this function.

To accomplish this the RIC utilizes a dedicated hub management interface. This is similar to the Inter-RIC bus since it allows the data packet to be recovered from the receiving RIC. Unlike the Inter-RIC bus the intended recipient is not another RIC but National Semiconductor's DP83932 "SONIC™" Network controller. The use of a dedicated bus allows a management status field to be appended at the end of the data packet. This can be done without affecting the operation of the repeater system.

### Processor Interface

The RIC's processor interface allows connection to a system processor. Data transfer occurs via an octal bi-directional data bus. The RIC has a number of on-chip registers indicating the status of the hub management functions, chip configuration and port status. These may be accessed by providing the chosen address at the Register Address (RA4-RA0) input pins.

Display update cycles and processor accesses occur utilizing the same data bus. An on-chip arbiter in the processor/display block schedules and controls the accesses and ensures the correct information is written into the display latches. During the display update cycles the RIC behaves as a master of its data bus. This is the default state of the data bus. Consequently, a TRI-STATE buffer must be placed between the RIC and the system processor's data bus. This

ensures bus contention is avoided during simultaneous display update cycles and processor accesses of other devices on the system bus. When the processor accesses a RIC register, the RIC enables the data buffer and selects the operation, either input or output, of the data pins.

### 5.2 DESCRIPTION OF REPEATER OPERATIONS

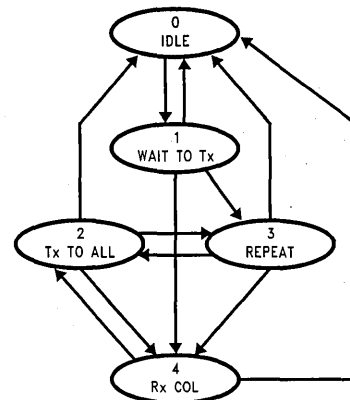
In order to implement a multi-chip repeater system which behaves as though it were a single logical repeater, special consideration must be paid to the data path used in packet repetition. For example, where in the path are specific operations such as Manchester decoding and elasticity buffering performed. Also the system's state machines which utilize available network activity signals, must be able to accommodate the various packet repetition and collision scenarios detailed in the repeater specification.

The RIC contains two types of inter-acting state machines. These are:

1. Port State Machines (PSMs). Every network attachment has its own PSM.
2. Main State Machine (MSM). This state machine controls the shared functional blocks as shown in the block diagram *Figure 5.1*.

### Repeater Port and Main State Machines

These two state machines are described in the following sections. Reference is made to expressions used in the IEEE Repeater specification. For the precise definition of these terms please refer to the specification. To avoid confusion with the RIC's implementation, where references are made to repeater states or terms as described in the IEEE specification, these items are written in *italics*. The IEEE state diagram is shown in *Figure 5-3*, the Inter-RIC bus state diagram is shown in *Figure 5-2*.



TL/F/11096-7

FIGURE 5.2. Inter-RIC Bus State Diagram

5.0 Functional Description (Continued)

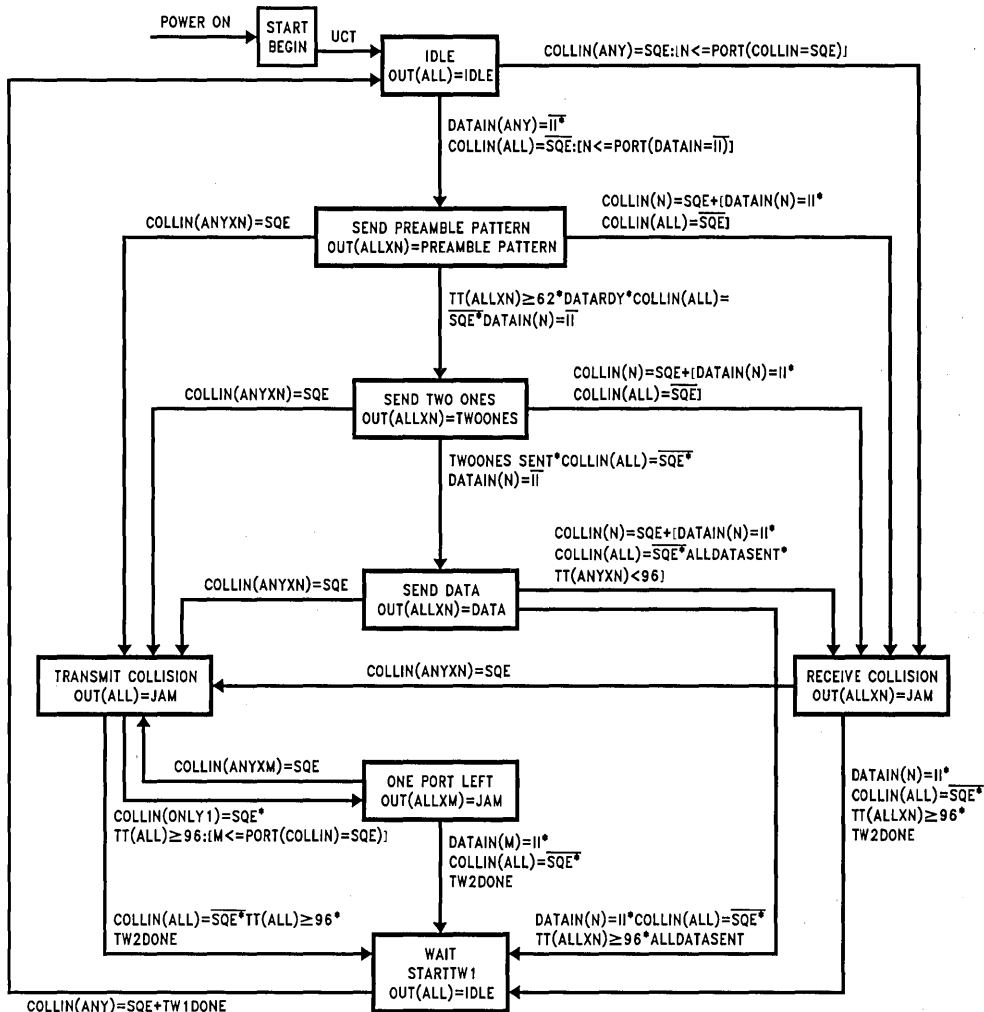


FIGURE 5.3. IEEE Repeater Main State Diagram

TL/F/11096-8

## 5.0 Functional Description (Continued)

### Port State Machine (PSM)

There are two primary functions for the PSM as follows:

1. Control the transmission of repeated data and jam signals over the attached segment.
2. Decide whether a port will be the source of data or collision information which will be repeated over the network. This repeater port is known as *PORT N*. An arbitration process is required to enable the repeater to transition from the *IDLE* state to the *SEND PREAMBLE PATTERN* or *RECEIVE COLLISION* states, see *Figure 5.3*. This process is used to locate the port which will be *PORT N* for that particular packet. The data received from this port is directed to the PLL decoder and transmitted over the Inter-RIC bus. If the repeater enters the *TRANSMIT COLLISION* state a further arbitration operation is performed to determine which port is *PORT M*. *PORT M* is differentiated from the repeater's other ports if the repeater enters the *ONE PORT LEFT* state. In this state *PORT M* does not transmit to its segment; where as all other ports are still required to transmit to their segments.

### Main State Machine (MSM)

The MSM controls the operation of the shared functional blocks in each RIC as shown in the block diagram, *Figure 5.1*, and it performs the majority of the data and collision propagation operations as defined by the IEEE specification, these include:

| Function                          | Action                                                                                                                                                                                 |
|-----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Preamble Regeneration             | Restore the length of the preamble pattern to the defined size.                                                                                                                        |
| Fragment Extension                | Extend received data or collision fragments to meet the minimum fragment length of 96 bits.                                                                                            |
| Elasticity Buffer Control         | A portion of the received packet may require storage in an Elasticity Buffer to accommodate preamble regeneration.                                                                     |
| Jam/Preamble Pattern Generation   | In cases of receive or transmit collisions a RIC is required to transmit a jam pattern (1010 . . . ).<br><b>Note:</b> This pattern is the same as that used for preamble regeneration. |
| Transmit Collision Enforcement    | Once the <i>TRANSMIT COLLISION</i> state is entered a repeater is required to stay in this state for at least 96 network bit times.                                                    |
| Data Encoding Control             | NRZ format data from the elasticity buffer must be encoded into Manchester format data prior to retransmission.                                                                        |
| <i>T<sub>w1</sub></i> Enforcement | Enforce the Transmit Recovery Time specification.                                                                                                                                      |
| <i>T<sub>w2</sub></i> Enforcement | Enforce Carrier Recovery Time specification on all ports with active collisions.                                                                                                       |

The interaction of the main and port state machines is visible, in part, by observing the Inter-RIC bus.

### Inter-RIC Bus Operation

#### Overview

The Inter-RIC Bus consists of eight signals. These signals implement a protocol which may be used to connect multiple RICs together. In this configuration, the logical function of a single repeater is maintained. The resulting multi-RIC system is compliant to the IEEE 802.3 repeater specification and may connect several hundred network segments. An example of a multi-RIC system is shown in *Figure 5.4*.

The Inter-RIC Bus connects multiple RICs to realize the following operations:

- Port N* Identification (which port the repeater receives data from)
- Port M* Identification (which port is the last one experiencing a collision)
- Data Transfer
- RECEIVE COLLISION* identification
- TRANSMIT COLLISION* identification
- DISABLE OUTPUT* (jabber protection)

The following tables briefly describes the operation of each bus signal, the conditions required for a RIC to assert a signal and which RICs (in a multi-RIC system) would monitor a signal:

| ACK <sub>I</sub>                                   |                                                                                                                                                                                                                                                                                |
|----------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Function                                           | Input signal to the PSM arbitration chain. This chain is employed to identify <i>PORT N</i> and <i>PORT M</i> .<br><b>Note:</b> A RIC which contains <i>PORT N</i> or <i>PORT M</i> may be identified by its $\overline{ACKO}$ signal being low when its $ACKI$ input is high. |
| Conditions required for a RIC to drive this signal | Not applicable                                                                                                                                                                                                                                                                 |
| RIC Receiving the signal                           | This is dependent upon the method used to cascade RICs, described in a following section.                                                                                                                                                                                      |

| $\overline{ACKO}$                                  |                                                                                           |
|----------------------------------------------------|-------------------------------------------------------------------------------------------|
| Function                                           | Output signal from the PSM arbitration chain.                                             |
| Conditions required for a RIC to drive this signal | This is dependent upon the method used to cascade RICs, described in a following section. |
| RIC Receiving the Signal                           | Not applicable                                                                            |

## 5.0 Functional Description (Continued)

| ACTN                                                      |                                                                                                                                                                                    |
|-----------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Function</b>                                           | This signal denotes there is activity on <i>PORT N</i> or <i>PORT M</i> .                                                                                                          |
| <b>Conditions required for a RIC to drive this signal</b> | A RIC must contain <i>PORT N</i> or <i>PORT M</i> .<br>Note: Although this signal normally has only one source asserting the signal active it is used in a wired-or configuration. |
| <b>RIC Receiving the Signal</b>                           | The signal is monitored by all RICs in the repeater system.                                                                                                                        |

| ANYXN                                                     |                                                                                                                  |
|-----------------------------------------------------------|------------------------------------------------------------------------------------------------------------------|
| <b>Function</b>                                           | This signal denotes that a repeater port that is not <i>PORT N</i> or <i>PORT M</i> is experiencing a collision. |
| <b>Conditions required for a RIC to drive this signal</b> | Any RIC which satisfies the above condition.<br>Note: This bus line is used in a wired-or configuration.         |
| <b>RIC Receiving the Signal</b>                           | The signal is monitored by all RICs in the repeater system.                                                      |

| COLN                                                      |                                                                     |
|-----------------------------------------------------------|---------------------------------------------------------------------|
| <b>Function</b>                                           | Denotes <i>PORT N</i> or <i>PORT M</i> is experiencing a collision. |
| <b>Conditions required for a RIC to drive this signal</b> | A RIC must contain <i>PORT N</i> or <i>PORT M</i> . (Note 1)        |
| <b>RIC Receiving the Signal</b>                           | The Signal is monitored by all other RICs in the repeater system.   |

| IRE                                                       |                                                                             |
|-----------------------------------------------------------|-----------------------------------------------------------------------------|
| <b>Function</b>                                           | This signal acts as an activity framing signal for the IRC and IRD signals. |
| <b>Conditions required for a RIC to drive this signal</b> | A RIC must contain <i>PORT N</i> .                                          |
| <b>RIC Receiving the Signal</b>                           | The Signal is monitored by all other RICs in the repeater system.           |

Note 1: Refer to note on page 25 for the transmit collision case.

| IRD                                                       |                                                                                                   |
|-----------------------------------------------------------|---------------------------------------------------------------------------------------------------|
| <b>Function</b>                                           | Decoded serial data, in NRZ format, received from the network segment attached to <i>PORT N</i> . |
| <b>Conditions required for a RIC to drive this signal</b> | A RIC must contain <i>PORT N</i> .                                                                |
| <b>RIC Receiving the Signal</b>                           | The signal is monitored by all other RICs in the repeater system.                                 |

| IRC                                                       |                                                                   |
|-----------------------------------------------------------|-------------------------------------------------------------------|
| <b>Function</b>                                           | Clock signal associated with IRD and IRE.                         |
| <b>Conditions required for a RIC to drive this signal</b> | A RIC must contain <i>PORT N</i> .                                |
| <b>RIC Receiving the Signal</b>                           | The signal is monitored by all other RICs in the repeater system. |

### Methods of RIC Cascading

In order to build multi-RIC repeaters *PORT N* and *PORT M* identification must be performed across all the RICs in the system. Inside each RIC the PSMs are arranged in a logical arbitration chain where port 1 is the highest and port 13 the lowest. The top of the chain, the input to port 1 is accessible to the user via the RIC's  $\overline{\text{ACKI}}$  input pin. The output from the bottom of the chain becomes the  $\overline{\text{ACKO}}$  output pin. In a single RIC system *PORT N* is defined as the highest port in the arbitration chain with receive or collision activity. *Port N* identification is performed when the repeater is in the *IDLE* state. *PORT M* is defined as the highest port in the chain with a collision when the repeater leaves the *TRANSMIT COLLISION* state. In order for the arbitration chain to function, all that needs to be done is to tie the  $\overline{\text{ACKI}}$  signal to a logic high state. In multi-RIC systems there are two methods to propagate the arbitration chain between RICs:

The first and most straight forward is to extend the arbitration chain by daisy chaining the  $\overline{\text{ACKI}}$   $\overline{\text{ACKO}}$  signals between RICs. In this approach one RIC is placed at the top of the chain (its  $\overline{\text{ACKI}}$  input is tied high), then the  $\overline{\text{ACKO}}$  signal from this RIC is sent to the  $\overline{\text{ACKI}}$  input of the next RIC and so on. This arrangement is simple to implement but it places some topological restrictions upon the repeater system. In particular, if the repeater is constructed using a backplane with removable printed circuit boards. (These boards contain the RICs and their associated components). If one of the boards is removed then the  $\overline{\text{ACKI}}$   $\overline{\text{ACKO}}$  chain will be broken and the repeater will not operate correctly.

## 5.0 Functional Description (Continued)

The second method of *PORT N* or *M* identification avoids this problem. This second technique relies on an external parallel arbiter which monitors all of the RIC's  $\overline{ACKO}$  signals and responds to the RIC with the highest priority. In this scheme each RIC is assigned with a priority level. One method of doing this is to assign a priority number which reflects the position of a RIC board on the repeater backplane, i.e., its slot number. When a RIC experiences receive activity and the repeater system is in the *IDLE* state, the RIC board will assert  $\overline{ACKO}$ . External arbitration logic drives the identification number onto an arbitration bus and the RIC containing *PORT N* will be identified. An identical procedure is used in the *TRANSMIT COLLISION* state to identify *PORT M*. This parallel means of arbitration is not subject to the problems caused by missing boards, i.e., empty slots in the backplane. The logic associated with asserting this arbitration vector in the various packet repetition scenarios could be implemented in programmable logic type devices.

To perform *PORT N* or *M* arbitration both of the above methods employ the same signals:  $\overline{ACKI}$ ,  $\overline{ACKO}$  and *ACTN*. The Inter-RIC bus allows multi-RIC operations to be performed in exactly the same manner as if there is only a single RIC in the system. The simplest way to describe the operation of Inter-RIC bus is to see how it is used in a number of common packet repetition scenarios. Throughout this description the RICs are presumed to be operating in external transceiver mode. This is advantageous for the explanation since the receive, transmit and collision signals from each network segment are observable. In internal transceiver mode this is not the case, since the collision signal for the non-AUI ports is derived by the transceivers inside the RIC.

### 5.3 EXAMPLES OF PACKET REPETITION SCENARIOS

#### Data Repetition

The simplest packet operation performed over the Inter-RIC Bus is data repetition. In this operation a data packet is received at one port and transmitted to all other segments.

The first task to be performed is *PORT N* identification. This is an arbitration process performed by the Port State Machines in the system. In situations where two or more ports simultaneously receive packets the Inter-RIC bus operates by choosing one of the active ports and forcing the others to transmit data. This is done to faithfully follow the IEEE specification's allowed exit paths from the *IDLE* state, i.e., to the *SEND PREAMBLE PATTERN* or *RECEIVE COLLISION* states.

The packet begins with a preamble pattern derived from the RIC's on chip jam/preamble generator. The data received at *PORT N* is directed through the receive multiplexor to the

PLL decoder. Once phase lock has been achieved, the decoded data, in NRZ format, with its associated clock and enable signals are asserted onto the IRD IRE and IRC Inter-RIC bus lines. This serial data stream is received from the bus by all RICs in the repeater and directed to their Elasticity Buffers. Logic circuits monitor the data stream and look for the Start of Frame Delimiter (SFD). When this has been detected data is loaded into the elasticity buffer for later transmission. This will occur when sufficient preamble has been transmitted and certain internal state machine operations have been fulfilled.

Figure 5.4 shows two RICs A and B, daisy chained together with RIC A positioned at the top of the chain. A packet is received at port B1 of RIC B and is then repeated by the other ports in the system. Figure 5.5 shows the functional timing diagram for this packet repetition represented by the signals shown in Figure 5.4. In this example only two ports in the system are shown, obviously the other ports also repeat the packet. It also indicates the operation of the RICs' state machines in so far as can be seen by observing the Inter-RIC bus. For reference, the repeater's state transitions are shown in terms of the states defined by the IEEE specification. The location, i.e., which port it is, of *PORT N* is also shown. The following section describes the repeater and Inter-RIC bus transitions shown in Figure 5.5.

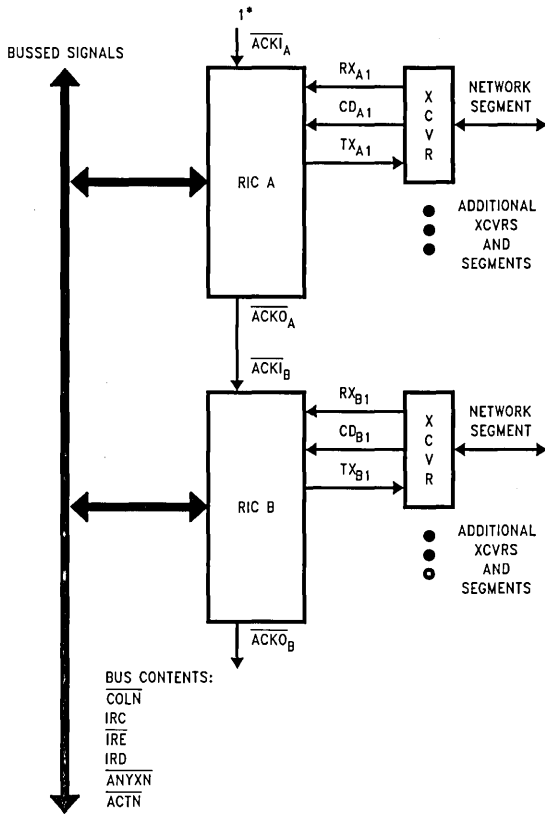
The repeater is stimulated into activity by the data signal received by port B1. The RICs in the system are alerted to forthcoming repeater operation by the falling edges on the  $\overline{ACKI}$   $\overline{ACKO}$  daisy chain and the *ACTN* bus signal. Following a defined start up delay the repeater moves to the *SEND PREAMBLE* state. The RIC system utilizes the start up delay to perform port arbitration. When packet transmission begins the RIC system enter the *REPEAT* state.

The expected, for normal packet repetition, sequence of repeater states, *SEND PREAMBLE*, *SEND SFD* and *SEND DATA* is followed but is not visible upon the Inter-RIC bus. They are merged together into a single *REPEAT* state. This is also true for the *WAIT* and *IDLE* states, they appear as a combined Inter-RIC bus *IDLE* state.

Once a repeat operation has begun, i.e., the repeater leaves the *IDLE* state. It is required to transmit at least 96 bits of data or jam/preamble onto its network segments. If the duration of the received signal from *PORT N* is smaller than 96 bits, the repeater transitions to the *RECEIVE COLLISION* state (described later). This behavior is known as fragment extension.

After the packet data has been repeated, including the emptying of the RICs' elasticity buffers, the RIC performs the *T<sub>w1</sub>* transmit recovery operation. This is performed during the *WAIT* state shown in the repeater state diagram.

### 5.0 Functional Description (Continued)

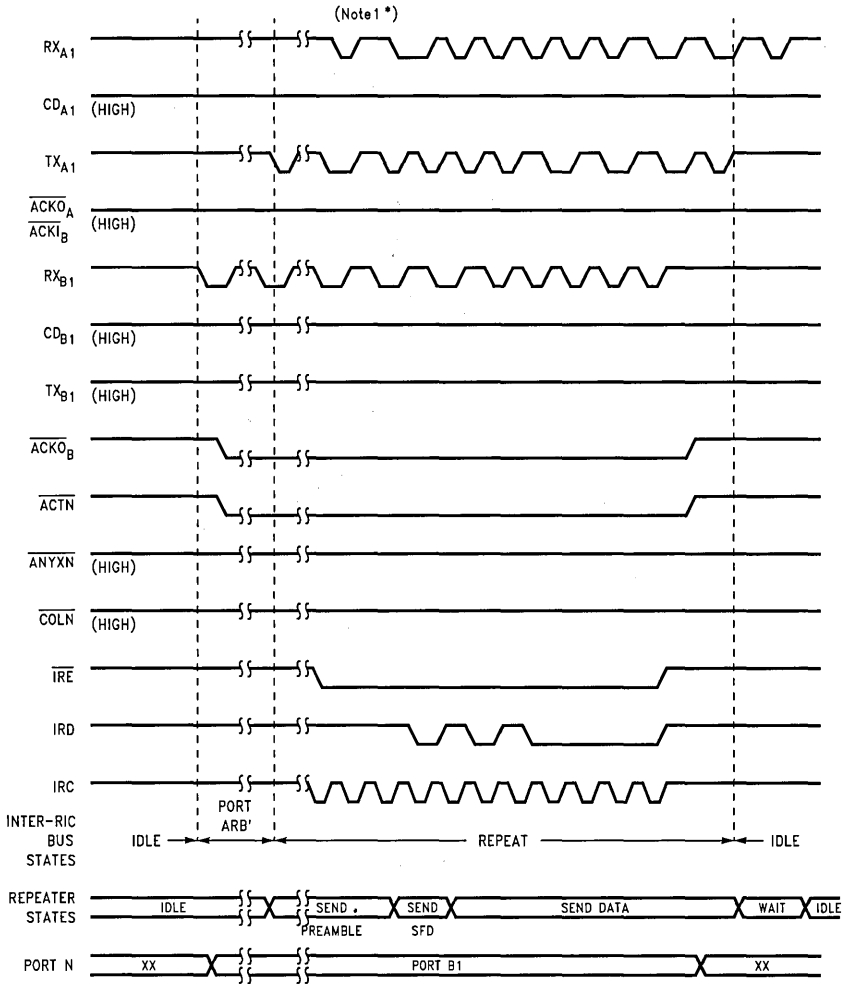


Note: In this example the Inter-RIC bus is configured to use active low signals.

TL/F/11096-9

**FIGURE 5.4. RIC System Topology**

### 5.0 Functional Description (Continued)



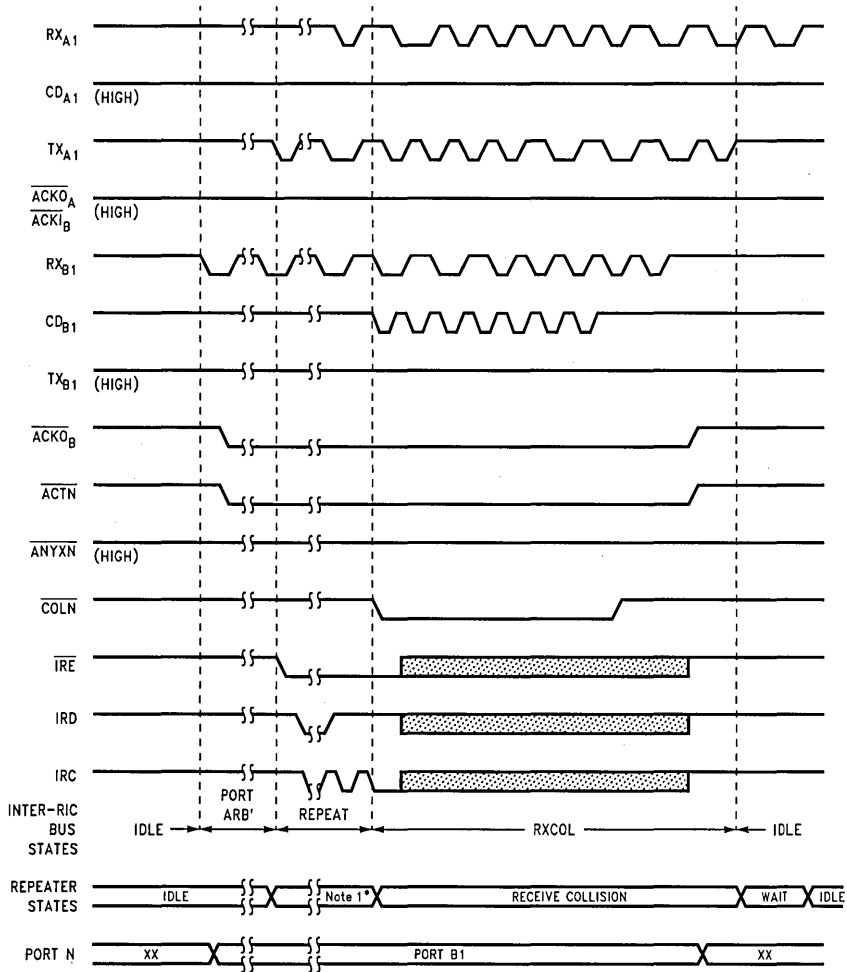
**\*Note 1:** The activity shown in  $RX_{A1}$  represents the transmitted signal on  $TX_{A1}$  after being looped back by the attached transceiver.  
**Note:** In this example the Inter-RIC bus is configured to use active low signals.

TL/F/11096-10

**FIGURE 5.5. Data Repetition**



### 5.0 Functional Description (Continued)



**\*Note 1:** SEND PREAMBLE, SEND SFD, SEND DATA.

TL/F/11096-11

**Note:** In this example the Inter-RIC bus is configured to use active low signals.

**FIGURE 5.6. Receive Collision**

## 5.0 Functional Description (Continued)

### Receive Collisions

A receive collision is a collision which occurs on the network segment attached to *PORT N*, i.e., the collision is "received" in a similar manner as a data packet is received and then repeated to the other network segments. Not surprisingly receive collision propagation follows a similar sequence of operations as is found with data repetition:

An arbitration process is performed to find *PORT N* and a preamble/jam pattern is transmitted by the repeater's other ports. When *PORT N* detects a collision on its segment the COLN Inter-RIC bus signal is asserted. This forces all the RICs in the system to transmit a preamble/jam pattern to their segments. This is important since they may be already transmitting data from their elasticity buffers. The repeater moves to the *RECEIVE COLLISION* state when the RICs begin to transmit the jam pattern. The repeater remains in this state until both the following conditions have been fulfilled:

1. At least 96 bits have been transmitted onto the network,
2. The activity has ended.

Under close examination the repeater specification reveals that the actual end of activity has its own permutations of conditions:

1. Collision and receive data signals may end simultaneously,
2. Receive data may appear to end before collision signals,
3. Receive data may continue for some time after the end of the collision signal.

Network segments using coaxial media may experience spurious gaps in segment activity when the collision signal goes inactive. This arises from the inter-action between the receive and collision signal squelch circuits, implemented in coaxial transceivers, and the properties of the coaxial cable itself. The repeater specification avoids propagation of these activity gaps by extending collision activity by the *Tw2* wait time. Jam pattern transmission must be sustained throughout this period. After this, the repeater will move to the *WAIT* state unless there is a data signal being received by *PORT N*.

The functional timing diagram, *Figure 5.6*, shows the operation of a repeater system during a receive collision. The system configuration is the same as earlier described and is shown in *Figure 5.4*.

The RICs perform the same *PORT N* arbitration and data repetition operations as previously described. The system is notified of the receive collision on port B1 by the COLN bus signal going active. This is the signal which informs the main state machines to output the jam pattern rather than the data held in the elasticity buffers. Once a collision has occurred the IRC, IRD AND IRE bus signals may become undefined. When the collision has ended and the *Tw2* operation performed, the repeater moves to the *WAIT* state.

### Transmit Collisions

A transmit collision is a collision that is detected upon a segment to which the repeater system is transmitting. The port state machine monitoring the colliding segment asserts the ANYXN bus signal. The assertion of ANYXN causes *PORT M* arbitration to begin. The repeater moves to the

*TRANSMIT COLLISION* state when the port which has been *PORT N* starts to transmit a Manchester encoded 1 on to its network segment. Whilst in the *TRANSMIT COLLISION* state all ports of the repeater must transmit the 1010 . . . jam pattern and *PORT M* arbitration is performed. Each RIC is obliged, by the IEEE specification, to ensure all of its ports transmit for at least 96 bits once the *TRANSMIT COLLISION* state has been entered. This transmit activity is enforced by the ANYXN bus signal. Whilst ANYXN is active all RIC ports will transmit jam. To ensure this situation lasts for at least 96 bits, the MSMs inside the RICs assert the ANYXN signal throughout this period. After this period has elapsed, ANYXN will only be asserted if there are multiple ports with active collisions on their network segments.

There are two possible ways for a repeater to leave the *TRANSMIT COLLISION* state. The most straight forward is when network activity, i.e., collisions and their *Tw2* extensions, end before the 96 bit enforced period expires. Under these conditions the repeater system may move directly to the *WAIT* state when 96 bits have been transmitted to all ports. If the MSM enforced period ends and there is still one port experiencing a collision the *ONE PORT LEFT* state is entered. This may be seen on the Inter-RIC bus when ANYXN is deasserted and *PORT M* stops transmitting to its network segment. In this circumstance the Inter-RIC bus transitions to the *RECEIVE COLLISION* state. The repeater will remain in this state whilst *PORT M's* collision, *Tw2* collision extension and any receive signals are present. When these conditions are not true, packet repetition finishes and the repeater enters the *WAIT* state.

*Figure 5.7* shows a multi-RIC system operating under transmit collision conditions. There are many different scenarios which may occur during a transmit collision, this figure illustrates one of these. The diagram begins with packet reception by port A1. Port B1 experiences a collision, since it is not *PORT N* it asserts ANYXN. This alerts the main state machines in the system to switch from data to jam pattern transmission.

Port A1 is also monitoring the ANYXN bus line. Its assertion forces A1 to relinquish its *PORT N* status, start transmitting, stop asserting ACTN and release its hold on the PSM arbitration signals (*ACKO A* and *ACKI B*). The first bit it transmit will be a Manchester encoded "1" in the jam pattern. Since port B1 is the only port with a collision it attains *PORT M* status and stops asserting ANYXN. It does however assert ACTN, and exert its presence upon the PSM arbitration chain (forces *ACKO B* low). The MSMs ensure that ANYXN stays active and thus force all of the ports, including *PORT M*, to transmit to their segments.

After some time port A1 experiences a collision. This arises from the presence of the packet being received from port A1's segment and the jam signal the repeater is now transmitting onto this segment. Two packets on one segment results in a collision. *PORT M* now moves from B1 to A1. Port A1 fulfills the same criteria as B1, i.e., it has an active collision on its segment, but in addition it is higher in the arbitration chain. This priority yields no benefits for port A1 since the ANYXN signal is still active. There are now two sources driving ANYXN, the MSMs and the collision on port B1.

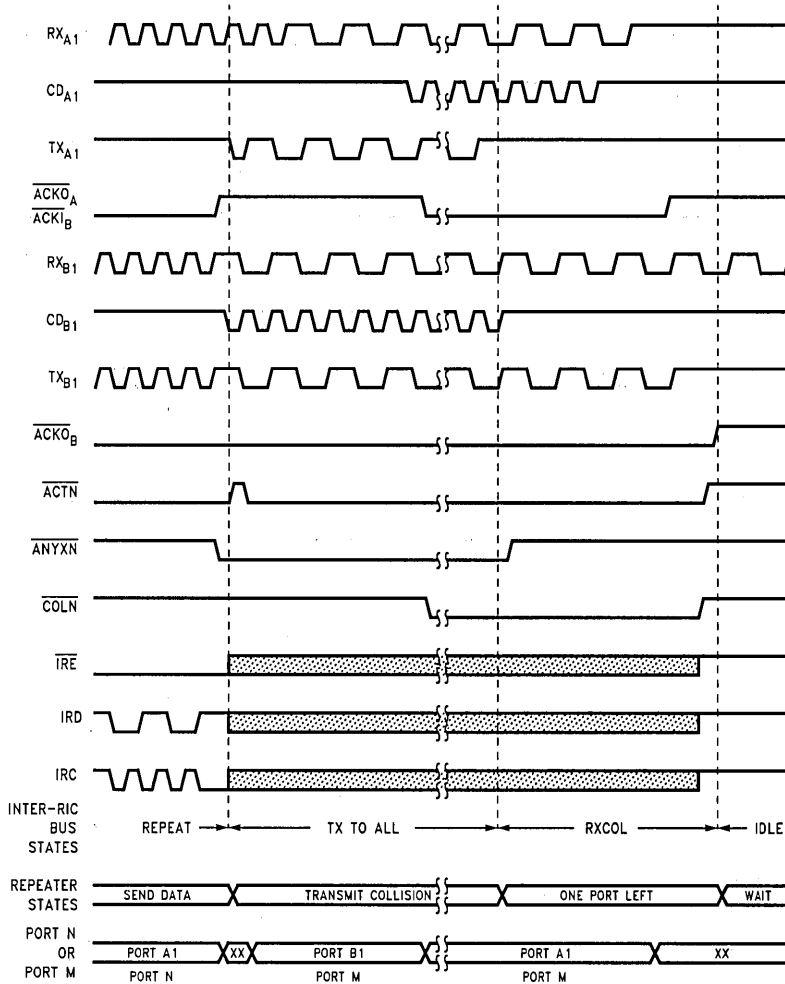
Eventually the collision on port B1 ends and the ANYXN extension by the MSMs expires. There is only one collision

### 5.0 Functional Description (Continued)

on the network (this may be deduced since ANYXN is inactive) so the repeater will move to the *ONE PORT LEFT* state. The RIC system treats this state in a similar manner to a receive collision with *PORT M* fulfilling the role of the receiving port. The difference from a true receive collision is that the switch from packet data to the jam pattern has already been made (controlled by ANYXN). Thus the state of COLN has no effect upon repeater operations. In com-

mon with the operation of the *RECEIVE COLLISION* state, the repeater remains in this condition until the collision and receive activity on *PORT M* subsides. The packet repetition operation completes when the *TW1* recovery time in the *WAIT* state has been performed.

**Note:** In transmit collision conditions COLN will only go active if the RIC which contained *PORT N* at the start of packet repetition contains *PORT M* during the *TRANSMIT COLLISION* and *ONE PORT LEFT* states.



**Note:** In this example the Inter-RIC bus is configured to use active low signals.

**FIGURE 5.7. Transmit Collision**

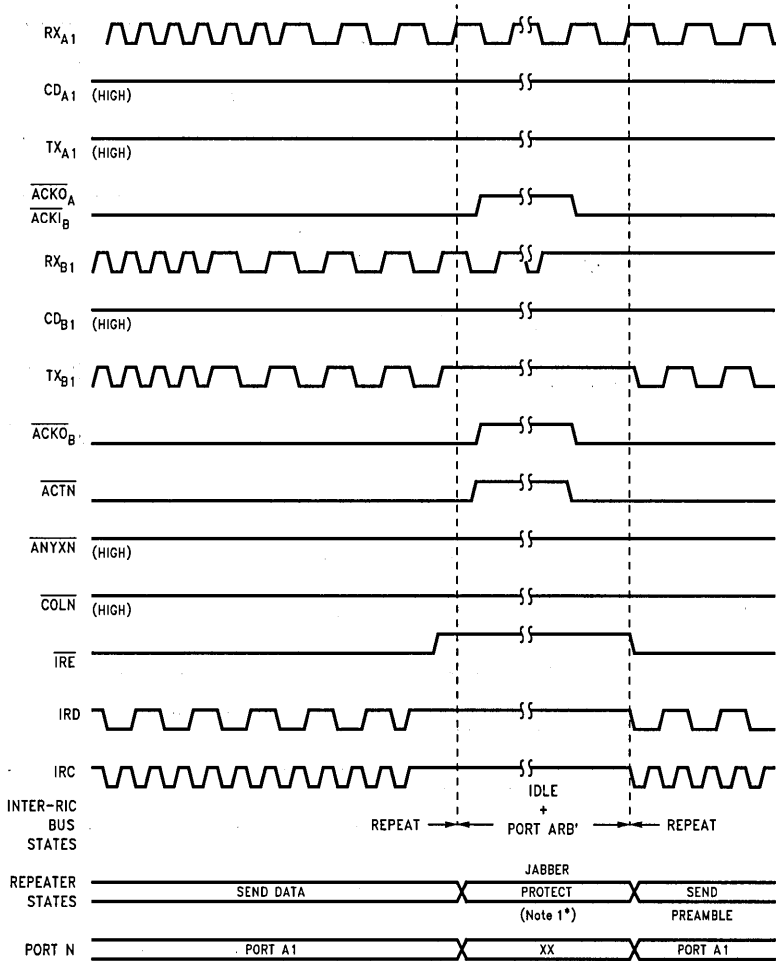
TL/F/11096-12

## 5.0 Functional Description (Continued)

### Jabber Protection

A repeater is required to disable transmit activity if the length of its current transmission reaches the jabber protect limit. This is defined by the specification's  $T_{w3}$  time. The repeater disables output for a time period defined by the  $T_{w4}$  specification, after this period normal operation may resume.

Figure 5.8 shows the effect of a jabber length packet upon a RIC based repeater system. The **JABBER PROTECT** state is entered from the **SEND DATA** state. While the  $T_{w4}$  period is observed the Inter-RIC bus displays the IDLE state. This is misleading since new packet activity or continuous activity (as shown in the diagram) does not result in packet repetition. This may only occur when the  $T_{w4}$  requirement has been satisfied.



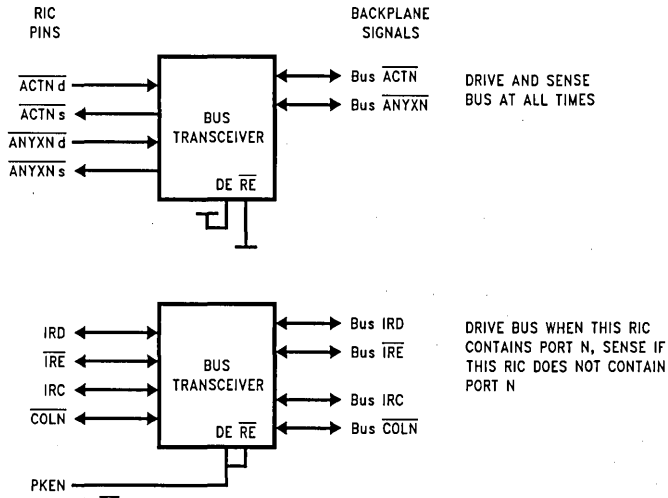
TL/F/11096-13

\*Note 1: The IEEE Specification does not have a jabber protect state defined in its main state diagram, this behaviour is defined in an additional MAU Jabber Lockup Protection state diagram.

Note: In this example the Inter-RIC bus is configured to use active low signals.

FIGURE 5.8. Jabber Protect

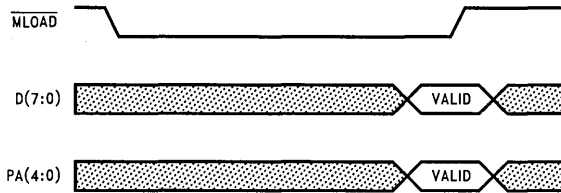
### 5.0 Functional Description (Continued)



**Note:** DE = Bus Drive Enable Active High, RE = Bus Receive Enable active low.  
**Note:** In this example the Inter-RIC bus is shown as using active low signals.

TL/F/11096-14

**FIGURE 5.9. External Bus Transceiver Connection Diagram**



TL/F/11096-15

**FIGURE 5.10. Mode Load Operation**

## 5.0 Functional Description (Continued)

### 5.4 DESCRIPTION OF HARDWARE CONNECTION FOR INTER-RIC BUS

When considering the hardware interface the Inter-RIC bus may be viewed as consisting of three groups of signals:

1. Port Arbitration chain, namely:  $\overline{ACKI}$  and  $\overline{ACKO}$ .
2. Simultaneous drive and sense signals, i.e., ACTN and ANYXN. (Potentially these signals may be driven by multiple devices).
3. Drive or sense signals, i.e., IRE, IRD, IRC and COLN. (Only one device asserts these signals at any instance in time.)

The first set of signals are either used as point to point links or with external arbitration logic. In both cases the load on these signals will not be large so that the on-chip drivers are adequate. This may not be true for signal classes (2) and (3).

The Inter-RIC bus has been designed to connect RICs together directly or via external bus transceivers. The latter is advantageous in large repeaters. In the second application the backplane is often heavily loaded and is beyond the drive capability of the on-chip bus drivers. The need for simultaneous sense and drive capabilities on the ACTN and ANYXN signals and the desire to allow operation with external bus transceivers makes it necessary for these bus signals to each have a pair of pins on the RIC. One driving the bus the other sensing the bus signal. When external bus transceivers are used they must be open collector/open drain to allow wire-ORing of the signals. Additionally, the drive and sense enables of the bus transceiver should be tied in the active state.

When the RIC is used in a stand alone configuration, it is required to tie  $ACTN_D$  to  $ACTN_S$  and  $ANYXN_D$  to  $ANYXN_S$ . The uni-directional nature of information transfer on the IRE, IRD, IRC and COLN signals, means a RIC is either driving these signals or receiving them from the bus but not both at the same time. Thus a single bi-directional input/output pin is adequate for each of these signals. In an external bus transceiver is used with these signals the Packet Enable "PKEN" RIC output pin performs the function of a drive enable and sense disable.

*Figure 5.9* shows the RIC connected to the Inter-RIC bus via external bus transceivers, such as National's DS3893A bus transceivers.

Some bus transceivers are of the inverting type. To allow the Inter-RIC bus to utilize these transceivers the RIC may

be configured to invert the active states of the ACTN, ANYXN, COLN and IRE signals. Instead of being active low they are active high.

Thus they become active low once more when passed through an inverting bus driver. This is particularly important for the ACTN and ANYXN bus lines, since these signals must be used in a wired-or configuration. Incorrect signal polarity would make the bus unusable.

### 5.5 PROCESSOR AND DISPLAY INTERFACE

The processor interface pins, which include the data bus, address bus and control signals, actually perform three operations which are multiplexed on these pins. These operations are:

1. The Mode Load Operation, which performs a power up initialization cycle upon the RIC.
2. Display Update Cycles, which are refresh operations for updating the display LEDs.
3. Processor Access Cycles, which allows  $\mu P$ 's to communicate with the RIC's registers.

These three operations are described below.

#### Mode Load Operation

The Mode Load Operation is a hardware initialization procedure performed at power on. It loads vital device configuration information into on-chip configuration registers. In addition to its configuration function the  $\overline{MLOAD}$  pin is the RIC's reset input. When  $\overline{MLOAD}$  is low all of the RIC's repeater timers, state machines, segment partition logic and hub management logic are reset.

The Mode Load Operation may be accomplished by attaching the appropriate set of pull up and pull down resistors to the data and register address pins to assert logic high or low signals onto these pins, and the providing a rising edge on the  $\overline{MLOAD}$  pin as is shown in *Figure 5.10*. The mapping of chip functions to the configuration inputs is shown in Table 5.1. Such an arrangement may be performed using a simple resistor, capacitor, diode network. Performing the Mode Load Operation in this way enables the configuration of a RIC that is in a simple repeater system (one without a processor).

Alternatively in a complex repeater system, the Mode Load Operation may be performed using a processor write cycle. This would require the  $\overline{MLOAD}$  pin be connected to the CPU's write strobe via some decoding logic, and included in the processor's memory map.

## 5.0 Functional Description (Continued)

TABLE 5.1. Pin Definitions for Options in the Mode Load Operation

| Pin Name | Programming Function       | Effect When Bit Is 0 | Effect When Bit Is 1 | Function                                                                                                                                                                                                                                                                                                                                                         |
|----------|----------------------------|----------------------|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0       | resv                       | Not Permitted        | Required             | To ensure correct device operation, this bit must be written with a logic one during the mode load operation.                                                                                                                                                                                                                                                    |
| D1       | tw2                        | 5 bits               | 3 bits               | This allows the user to select one of two values for the repeater specification tw2 time. The lower limit (3 bits) meets the IEEE specification. The upper limit (5 bits) is not specification compliant but may provide users with higher network throughput by avoiding spurious network activity gaps when using coaxial (10BASE2, 10BASE5) network segments. |
| D2       | $\overline{\text{CCLIM}}$  | 63                   | 31                   | The partition specification requires a port to be partitioned after a certain number of consecutive collisions. The RIC has two values available to allow users to customize the partitioning algorithm to their environment. Please refer to the Partition State Machine, in data sheet Section 7.3.                                                            |
| D3       | $\overline{\text{LPPART}}$ | Selected             | Not Selected         | The RIC may be configured to partition a port if the segment transceiver does not loopback data to the port when the port is transmitting to it, as described in the Partition State Machine.                                                                                                                                                                    |
| D4       | $\overline{\text{OWCE}}$   | Selected             | Not Selected         | This configuration bit allows the on-chip partition algorithm to include out of window collisions into the collisions it monitors, as described in the Partition State Machine.                                                                                                                                                                                  |
| D5       | $\overline{\text{TXONLY}}$ | Selected             | Not Selected         | This configuration bit allows the on-chip partition algorithm to restrict segment reconnection, as described in the Partition State Machine.                                                                                                                                                                                                                     |
| D6       | $\overline{\text{DPART}}$  | Selected             | Not Selected         | The Partition state machines for all ports may be disabled by writing a logic zero to this bit during the mode load operation.                                                                                                                                                                                                                                   |
| D7       | MIN/MAX                    | Minimum Mode         | Maximum Mode         | The operation of the display update block is controlled by the value of this configuration bit, as described in the Display Update Cycles section.                                                                                                                                                                                                               |

## 5.0 Functional Description (Continued)

**TABLE 5.1 Pin Definitions for Options in the Mode Load Operation (Continued)**

| Pin Name | Programming Function | Effect When Bit is 0                                                                     | Effect When Bit is 1 | Function                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
|----------|----------------------|------------------------------------------------------------------------------------------|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|--------|-------------|---|---|-------------------------------------------------------------|---|---|------------------------------------------------------------------------------------------|---|---|------------------------------------------------------------------------------------------|---|---|-------------------------------------------------------------|
| RA0      | BYPAS1               |                                                                                          |                      | <p>These configuration bits select which of the repeater ports (numbers 2 to 13) are configured to use the on-chip internal 10BASE-T transceivers or the external transceiver interface. The external transceiver interface operates using AUI compatible signal levels.</p> <table border="1"> <thead> <tr> <th>BYPAS2</th> <th>BYPAS1</th> <th>Information</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>All ports (2 to 13) use the external Transceiver Interface.</td> </tr> <tr> <td>0</td> <td>1</td> <td>Ports 2 to 7 use the external interface, 8 to 13 use the internal 10BASE-T transceivers.</td> </tr> <tr> <td>1</td> <td>0</td> <td>Ports 2 to 5 use the external interface, 6 to 13 use the internal 10BASE-T transceivers.</td> </tr> <tr> <td>1</td> <td>1</td> <td>All ports (2 to 13) use the internal 10BASE-T transceivers.</td> </tr> </tbody> </table> | BYPAS2 | BYPAS1 | Information | 0 | 0 | All ports (2 to 13) use the external Transceiver Interface. | 0 | 1 | Ports 2 to 7 use the external interface, 8 to 13 use the internal 10BASE-T transceivers. | 1 | 0 | Ports 2 to 5 use the external interface, 6 to 13 use the internal 10BASE-T transceivers. | 1 | 1 | All ports (2 to 13) use the internal 10BASE-T transceivers. |
| BYPAS2   | BYPAS1               | Information                                                                              |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
| 0        | 0                    | All ports (2 to 13) use the external Transceiver Interface.                              |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
| 0        | 1                    | Ports 2 to 7 use the external interface, 8 to 13 use the internal 10BASE-T transceivers. |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
| 1        | 0                    | Ports 2 to 5 use the external interface, 6 to 13 use the internal 10BASE-T transceivers. |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
| 1        | 1                    | All ports (2 to 13) use the internal 10BASE-T transceivers.                              |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
| RA1      | BYPAS2               |                                                                                          |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
| RA2      | BINV                 | Active High Signals                                                                      | Active Low Signals   | This selection determines whether the Inter-RIC signals: IRE, ACTN, ANYXN, COLN and Management bus signal MCRS are active high or low.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
| RA3      | EXPLL                | External PLL                                                                             | Internal PLL         | If desired, the RIC may be used with an external decoder, this configuration bit performs the selection.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
| RA4      | resv                 | Not Permitted                                                                            | Required             | To ensure correct device operation, this bit must be written with a logic one during the mode load operation.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |



## 5.0 Functional Description (Continued)

### 5.6 DESCRIPTION OF HARDWARE CONNECTION FOR PROCESSOR AND DISPLAY INTERFACE

#### Display Update Cycles

The RIC possesses control logic and interface pins which may be used to provide status information concerning activity on the attached network segments and the current status of repeater functions. These status cycles are completely autonomous and require only simple support circuitry to produce the data in a form suitable for a light emitting diode "LED" display. The display may be used in one of two modes:

1. Minimum Mode: General Repeater Status LEDs
2. Maximum Mode: Individual Port Status LEDs

Minimum mode, intended for simple LED displays, makes available four status indicators. The first LED denotes whether the RIC has been forced to activate its jabber protect functions. The remaining 3 LEDs indicate if any of the RIC's network segments are: (1) experiencing a collision, (2) receiving data, (3) currently partitioned. When minimum display mode is selected the only external components required are a 74LS374 type latch, the LEDs and their current limiting resistors.

Maximum mode differs from minimum mode by providing display information specific to individual network segments. This information denotes the collision activity, packet reception and partition status of each segment. In the case of 10BASE-T segments the link integrity status and polarity of the received data are also made available. The wide variety of information available in maximum mode may be used in its entirety or in part. Thus allowing the system designer to choose the appropriate complexity of status display commensurate with the specification of the end equipment.

The signals provided and their timing relationships have been designed to interface directly with 74LS259 type addressable latches. The number of latches used being dependant upon the complexity of the display. Since the latches are octal, a pair of latches is needed to display each type of segment specific data (13 ports means 13 latch bits). The accompanying tables (5.1 and 5.2) show the function of the interface pins in minimum and maximum modes. *Figure 5.12* shows the location of each port's status information when maximum mode is selected. This may be compared with the connection diagram *Figure 5.11*.

Immediately following the Mode Load Operation (when the MLOAD pin transitions to a high logic state), the display logic performs an LED test operation. This operation lasts one second and while it is in effect all of the utilized LEDs will blink on. Thus an installation engineer is able to test the operation of the display by forcing the RIC into a reset cycle (MLOAD forced low). The rising edge on the MLOAD pin starts the LED test cycle. **During the LED test cycle the RIC does not perform packet repetition operations.**

The status display possesses a capability to lengthen the time an LED is active. At the end of the repetition of a packet, the display is frozen showing the current activity. This freezing lasts for 30 milliseconds or until a subsequent packet is repeated. Thus at low levels of packet activity the display stretches activity information to make it discernable to the human eye. At high traffic rates the relative brightness of the LEDs indicates those segments with high or low activity.

It should be mentioned that when the Real Time Interrupt (RTI) occurs, the display update cycle will stop and after RTI is serviced, the display update cycle will resume activity.

TABLE 5.2. Status Display Pin Functions in Minimum Mode

| Signal Pin Name | Function in MINIMUM MODE                                                                                                                       |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| D0              | No operation                                                                                                                                   |
| D1              | Provides status information indicating if there is a collision occurring on one of the segments attached to this RIC.                          |
| D2              | Provides status information indicating if one of this RIC's ports is receiving a data or collision packet from a segment attached to this RIC. |
| D3              | Provides status information indicating that the RIC has experienced a jabber protect condition.                                                |
| D4              | Provides Status information indicating if one of the RIC's segments is partitioned.                                                            |
| D(7:5)          | No operation                                                                                                                                   |
| STR0            | This signal is the latch enable for the 374 type latch.                                                                                        |
| STR1            | This signal is held at a logic one.                                                                                                            |

## 5.0 Functional Description (Continued)

**Table 5.3 Status Display Pin Functions in MAXIMUM MODE**

| Signal Pin Name | Function in Maximum Mode                                                                                                                                                                                                                                                                               |
|-----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0              | Provides status information concerning the Link Integrity status of 10BASE-T segments. This signal should be connected to the data inputs of the chosen pair of 74LS259 latches.                                                                                                                       |
| D1              | Provides status information indicating if there is a collision occurring on one of the segments attached to this RIC. This signal should be connected to the data inputs of the chosen pair of 74LS259 latches.                                                                                        |
| D2              | Provides status information indicating if one of this RIC's ports is receiving a data or a collision packet from its segment. This signal should be connected to the data inputs of the chosen pair of 74LS259 latches.                                                                                |
| D3              | Provides Status information indicating that the RIC has experienced a jabber protect condition. Additionally it denotes which of its ports are partitioned. This signal should be connected to the data inputs of the chosen pair of 74LS259 latches.                                                  |
| D4              | Provides status information indicating if one of this RIC's ports is receiving data of inverse polarity. This status output is only valid if the port is configured to use its internal 10BASE-T transceiver. The signal should be connected to the data inputs of the chosen pair of 74LS259 latches. |
| D(7:5)          | These signals provide the repeater port address corresponding to the data available on D(4:0).                                                                                                                                                                                                         |
| STR0            | This signal is the latch enable for the lower byte latches, that is the 74LS259s which display information concerning ports 1 to 7.                                                                                                                                                                    |
| STR1            | This signal is the latch enable for the upper byte latches, that is the 74LS259s which display information concerning ports 8 to 13.                                                                                                                                                                   |

### Maximum Mode LED Definitions

#### 74LS259 Latch Inputs = STR0

| 259 Output      | Q0   | Q1      | Q2    | Q3    | Q4    | Q5    | Q6    | Q7    |
|-----------------|------|---------|-------|-------|-------|-------|-------|-------|
| 259 Addr S2-0   | 000  | 001     | 010   | 011   | 100   | 101   | 110   | 111   |
| RIC Port Number |      | 1 (AUI) | 2     | 3     | 4     | 5     | 6     | 7     |
| RIC D0 259 # 1  |      |         | LINK  | LINK  | LINK  | LINK  | LINK  | LINK  |
| RIC D1 259 # 2  | ACOL | COL     | COL   | COL   | COL   | COL   | COL   | COL   |
| RIC D2 259 # 3  | AREC | REC     | REC   | REC   | REC   | REC   | REC   | REC   |
| RIC D3 259 # 4  | JAB  | PART    | PART  | PART  | PART  | PART  | PART  | PART  |
| RIC D4 259 # 5  |      |         | BDPOL | BDPOL | BDPOL | BDPOL | BDPOL | BDPOL |

#### 74LS259 (or Equiv.) Latch Inputs = STR1

| 259 Output      | Q0    | Q1    | Q2    | Q3    | Q4    | Q5    | Q6  | Q7  |
|-----------------|-------|-------|-------|-------|-------|-------|-----|-----|
| 259 Addr S2-0   | 000   | 001   | 010   | 011   | 100   | 101   | 110 | 111 |
| RIC Port Number | 8     | 9     | 10    | 11    | 12    | 13    |     |     |
| RIC D0 259 # 6  | LINK  | LINK  | LINK  | LINK  | LINK  | LINK  |     |     |
| RIC D1 259 # 7  | COL   | COL   | COL   | COL   | COL   | COL   |     |     |
| RIC D2 259 # 8  | REC   | REC   | REC   | REC   | REC   | REC   |     |     |
| RIC D3 259 # 9  | PART  | PART  | PART  | PART  | PART  | PART  |     |     |
| RIC D4 259 # 10 | BDPOL | BDPOL | BDPOL | BDPOL | BDPOL | BDPOL |     |     |

This shows the LED Output Functions for the LED Drivers when 74LS259s are used. The top table refers to the bank of 4 74LS259s latched with STR0, and the lower table refers to the bank of 4 74LS259s latched with STR1. For example the RIC's D0 data signal goes to 259 # 1 and # 5. These two 74LS259s then drive the LINK LEDs).

Note: ACOL = Any Port Collision, AREC = Any Port Reception, JAB = Any Port Jabbering, LINK = Port Link, COL = Port Collision, REC = Port Reception, PART = Port Partitioned, BDPOL = Bad (inverse) Polarity or received data.

**FIGURE 5.12**

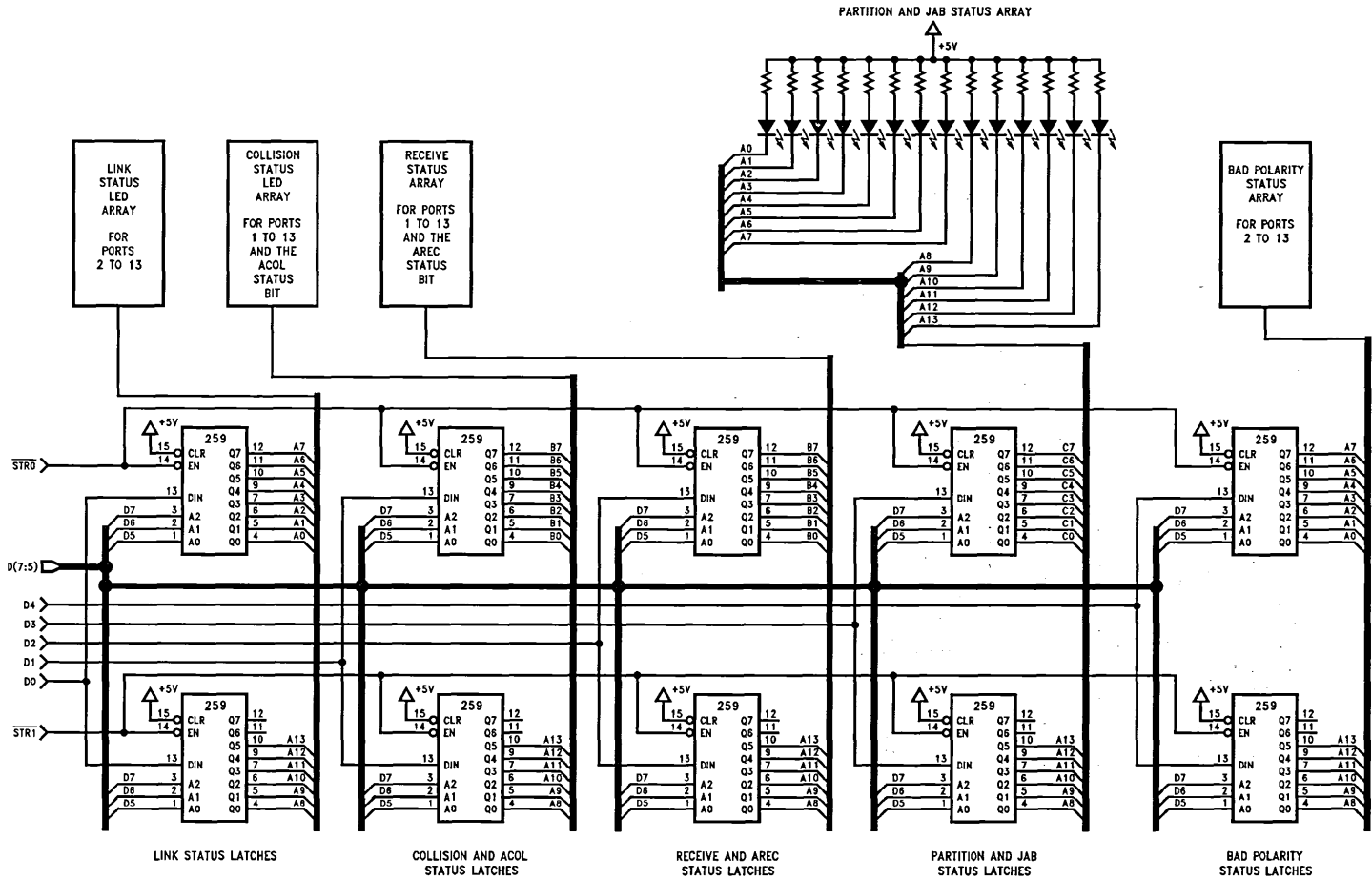


FIGURE 5.11. Maximum Mode LED Display (All Available Status Bits Used)

TL/F/11096-16

3-43



5.0 Functional Description (Continued)

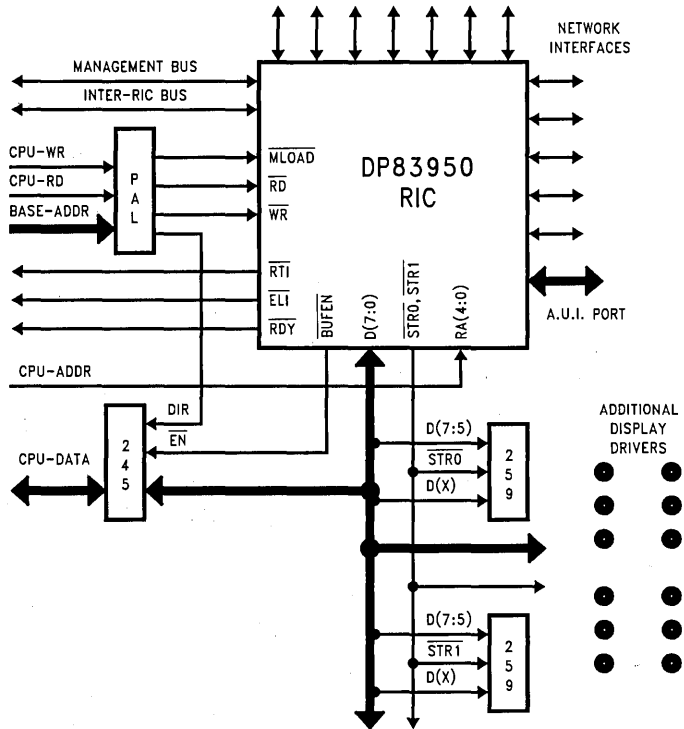


FIGURE 5.13. Processor Connection Diagram

TL/F/11096-17

## 5.0 Functional Description (Continued)

### Processor Access Cycles

Access to the RIC's on-chip registers is made via its processor interface. This utilizes conventional non-multiplexed address (five bit) and data (eight bit) busses. The data bus is also used to provide data and address information to off chip display latches during display update cycles. While performing these cycles the RIC behaves as a master of its data bus. Consequently a TRI-STATE bi-directional bus transceiver, e.g., 74LS245 must be placed between the RIC and any processor bus.

The processor requests a register access by asserting the read "RD" or write "WR" input strobes. The RIC responds by finishing any current display update cycle and asserts the tri-state buffer enable signal "BUFFEN". If the processor cycle is a write cycle then the RIC's data buffers are disabled to prevent contention. In order to interface to the RIC in a processor controlled system it is likely a PAL device will be used to perform the following operations:

1. Locate the RIC in the processor's memory map (address decode),
2. Generate the RIC's read and write strobes,
3. Control the direction signal for the 74LS245.

An example of the processor and display interfaces is shown in *Figure 5.13*.

## 6.0 Hub Management Support

The RIC provides information regarding the status of its ports and the packets it is repeating. This data is available in three forms:

1. Counted Events—Network events accumulated into the RIC's 16-bit Event Counter Registers.
2. Recorded Events—Network events that set bits in the Event Record Registers.
3. Hub Management Status Packets—This is information sent over the Management Bus in a serial function to be decoded by an Ethernet Controller board.

The counted and recorded event information is available through the processor interface. This data is port specific and may be used to generate interrupts via the Event Logging Interrupt "ELI" pin. Since the information is specific to each port, each repeater port has its own event record register and event counter. The counters and event record registers have user definable masks which enable them to be configured to count and record a variety of events. The counters and record registers are designed to be used together so that detailed information, i.e., a count value can be held on-chip for a specific network condition, and more general information, i.e., certain types of events have occurred, may be retained in on-chip latches. Thus the user may configure the counters to increment upon a rapidly occurring event (most likely to be used to count collisions), and the record registers may log the occurrence of less frequent error conditions such as jabber protect packets.

### 6.1 EVENT COUNTING FUNCTION

The counters may increment upon the occurrence of one of the categories of event as described below.

Potential sources for Counter increment:

**Jabber Protection (JAB):** The port counter increments if the length of a received packet from its associated port, causes the repeater state machine to enter the jabber protect state.

**Elasticity Buffer Error (ELBER):** The port counter increments if a Elasticity Buffer underflow or overflow occurs during packet reception. The flag is held inactive if a collision occurs during packet reception or if a phase lock error, described below, has already occurred during the repetition of the packet.

**Phase Lock Error (PLER):** A phase lock error is caused if the phase lock loop decoder loses lock during packet reception. Phase lock onto the received data stream may or may not be recovered later in the packet and data errors may have occurred. This flag is held inactive if a collision occurs.

**Non SFD Packet (NSFD):** If a packet is received and the start of frame delimiter is not found, the port counter will increment. Counting is inhibited if the packet suffers a collision.

**Out of Window Collision (OWC):** The out of window collision flag for a port goes active when a collision is experienced outside of the network slot time.

**Transmit Collision (TXCOL):** The transmit collision flag for a port is enabled when a transmit collision is experienced by the repeater. Each port experiencing a collision under these conditions is said to have suffered a transmit collision.

**Receive Collision (RXCOL):** The receive collision flag for a port goes active when the port is the receive source of network activity and suffers a collision, provided no other network segments experience collision then the receive collision flag for the receiving port will be set.

**Partition (PART):** The port counter increments when a port becomes partitioned.

**Bad Link (BDLNK):** The port counter increments when a port is configured for 10BASE-T operation has entered the link lost state.

**Short Event reception (SE):** The port counter increments if the received packet is less than 74 bits long and no collision occurs during reception.

**Packet Reception (REC):** When a packet is received the port counter increments.

In order to utilize the counters the user must choose, from the above list, the desired statistic for counting. This counter mask information must be written to the appropriate, Event Count Mask Register. There are two of these registers, the Upper and Lower, Event Count Mask registers. For the exact bit patterns of these registers please see Section 8 of the data sheet.

For example if the counters are configured to count network collisions and the appropriate masks have been set, then whenever a collision occurs on a segment, this information is latched by the hub management support logic. At the end of repetition of the packet the collision status, respective to each port, is loaded into that port's counter. This operation is completely autonomous and requires no processor intervention.

## 6.0 Hub Management Support (Continued)

Each counter is 16 bits long and may be directly read by the processor. Additionally each counter has a number of decodes to indicate the current value of the count. There are three decodes:

- Low Count (a value of 00FF Hex and under),
- High Count (a value of C000 Hex and above),
- Full Count (a value of FFFF Hex).

The decodes from each counter are logically "ORed" together and may be used as interrupt sources for the  $\overline{ELI}$  interrupt pin. Additionally the status of these bits may be observed by reading the Page Select Register (PSR), (see Section 8 for register details). In order to enable any of these threshold interrupts, the appropriate interrupt mask bit must be written to the Management and Interrupt Configuration Register; see Section 8 for register details.

In addition to their event masking functions the Upper Event Counting Mask Register (UECMR) possesses two bits which control the operation of the counters. When written to a logic one, the reset on read bit "ROR" resets the counter after a processor read cycle is performed. If this operation is not selected then in order to zero the counters they must either be written with zeros by the processor or allowed to roll over to all zeros. The freeze when full bit "FWF" prevents counter roll over by inhibiting count up cycles (these happen when chosen events occur), thus freezing the particular counter at FFFF Hex.

The port event counters may also be controlled by the Counter Decrement (CDEC) pin. As its name suggests a logic low state on this pin will decrement all the counters by a single value. The pulses on CDEC are internally synchronized and scheduled so as not to conflict with any "up counting" activity. If an up count and a down count occur simultaneously then the down count is delayed until the up count has completed. This combination of up and down counting capability enables the RIC's on-chip counters to provide a simple rolling average or be used as extensions of larger off chip counters.

**Note:** If the FWF option is enabled then the count down operation is disabled from those registers which have reached FFFF Hex and consequently have been frozen. Thus, if FWF is set and CDEC has been employed to provide a rate indication. A frozen counter indicates that a rate has been detected which has gone out of bounds, i.e., too fast increment or too slow increment. If the low count and high count decodes are employed as either interrupt sources or in a polling cycle, the direction of the rate excursion may be determined.

### Reading the Event Counters

The RIC's external data bus is eight bits wide, since the event counters are 16 bits long two processor read cycles are required to yield the counter value. In order to ensure that the read value is correct and to allow simultaneous event counts with processor accesses, a temporary holding register is employed. A read cycle to either the lower or upper byte of a counter, causes both bytes to be latched into the holding register. Thus when the other byte of the counter is obtained the holding register is accessed and not the actual counter register. This ensures that the upper and lower bytes contain the value sampled at the same instance in time, i.e., when the first read cycle to that counter occurred.

There is no restriction concerning whether the upper or lower byte is read first. However to ensure the "same instance value" is obtained, the reads of the upper then lower byte (or vice versa) should be performed as consecutive reads of

the counter array. Other NON COUNTER registers may be read in between these read cycles and also write cycles may be performed. If another counter is read or the same byte of the original counter is read, then the holding register is updated from the counter array and the unread byte is lost.

If the reset on read option is employed then the counter is reset after the transfer to the holding register is performed. Processor read and write cycles are scheduled in such a manner that they do not conflict with count up or count down operations. That is to say, in the case of a processor read the count value is stable when it is loaded into the holding register. In the case of a processor write, the newly written value is stable so it maybe incremented or decremented by any subsequent count operation. During the period the MLOAD pin is low, (power on reset) all counters are reset to zero and all count masks are forced into the disabled state. Section 8 of the data sheet details the address location of the port event counters.

### 6.2 EVENT RECORD FUNCTION

As previously stated each repeater port has its own Event Recording Register. This is an 8-bit status register each bit is dedicated to logging the occurrence of a particular event (see Section 8 for detailed description). The logging of these events is controlled by the Event Recording Mask Register, for an event to be recorded the particular mask bit must be set, (see Section 8 description of this register). Similar to the scheme employed for the event counters, the recorded events are latched during the repetition of a packet and then automatically loaded into the recording registers at the end of transmission of a packet. When one of the unmasked events occurs, the particular port register bit is set. This status is visible to the user. All of the register bits for all of the ports are logically "ORed" together to produce a Flag Found "FF" signal. This indicator may be found by reading the Page Select Register. Additionally an interrupt may be generated if the appropriate mask bit is enabled in the Management and Interrupt Configuration Register.

A processor read cycle to an Event Record Register resets any of the bits set in that register. Read operations are scheduled to guarantee non changing data during a read cycle. Any internal bit setting event which immediately follows a processor read will be successful. The events which may be recorded are described below:

**Jabber Protection (JAB):** This flag goes active if the length of a received packet from the relevant port, causes the repeater state machine to enter the Jabber Protect state.

**Elasticity Buffer Error (ELBER):** This condition occurs if an Elasticity Buffer full or overflow occurs during packet reception. The flag is held inactive if a collision occurs during packet reception or if a phase lock error has already occurred during the repetition of the packet.

**Phase Lock Error (PLER):** A phase lock error is caused if the phase lock loop decoder loses lock during packet reception. Phase lock onto the received data stream may or may not be recovered later in the packet and data errors may have occurred. This flag is held inactive if a collision occurs.

**Non SFD Packet (NSFD):** If a packet is received and the start of frame delimiter is not found, the flag will go active. The flag is held inactive if a collision occurs in during packet repetition.

## 6.0 Hub Management Support (Continued)

**Out of Window Collision (OWC):** The out of window collision flag for a port goes active when a collision is experienced outside of the network slot time.

**Partition (PART):** This flag goes active when a port becomes partitioned.

**Bad Link (BDLNK):** The flag goes active when a port is configured for 10BASE-T operation has entered the link lost state.

**Short Event reception (SE):** This flag goes active if the received packet is less than 74 bits long and no collision occurs during reception.

### 6.3 MANAGEMENT INTERFACE OPERATION

The HUB Management interface provides a mechanism to combine repeater status information with packet information to form a hub management status packet. The interface, a serial bus consisting of carrier sense, received clock and received data, is designed to connect one or multiple RIC's over a backplane bus to a DP83932 "SONIC" network controller. The SONIC and the RICs form a powerful entity for network statistics gathering.

The interface consists of four pins:

|       |                                                                        |
|-------|------------------------------------------------------------------------|
| MRXC  | Management Receive Clock—10 MHz NRZ Clock output.                      |
| MCRS  | Management Carrier Sense—Input/Output indicating of valid data stream. |
| MRXD  | Management Receive Data—NRZ Data output synchronous to MRXC.           |
| PCOMP | Packet Compress—Input to truncate the packet's data field.             |

The first three signals mimic the interface between an Ethernet controller and a phase locked loop decoder (specifically the DP83932 SONIC and DP83910 SNI), these signals are driven by the RIC receiving the packet. MRXC and MRXD compose an NRZ serial data stream compatible with the DP83932. The PCOMP signal is driven by logic on the processor board. The actual data stream transferred over MRXD is derived from data transferred over the IRD Inter-RIC bus line. These two data streams differ in two important characteristics:

1. At the end of packet repetition a hub management status field is appended to the data stream. This status field, consisting of 7 bytes is shown in *Figure 6.1* and *6.2*. The information field is obtained from a number of packet status registers described below. In common with the 802.3 protocol the least significant bit of a byte is transmitted first.
2. While the data field of the repeated packet is being transferred over the management bus, received clock signals on the MRXC pin may be inhibited. This operation is under the control of the Packet Compress pin PCOMP. If PCOMP is asserted during repetition of the packet then MRXC signals are inhibited when the number of bytes (after SFD) transferred over the management bus equals the number indicated in the Packet Compress Decode Register. This register provides a means to delay the effect of the PCOMP signal, which may be generated early in the packet's repetition, until the desired moment. Packet compression may be used to reduce the amount of

memory required to buffer packets when they are received and are waiting to be processed by hub management software. In this kind of application an address decoder, which forms part of the packet compress logic, would monitor the address fields as they are received over the management bus. If the destination address is not the address of the management node inside the hub, then packet compression could be employed. In this manner only the portion of the packet meaningful for hub management interrogation, i.e., the address fields, is transferred to the SONIC and is buffered in memory.

If the repeated packet ends before PCOMP is asserted or before the required number of bytes have been transferred, then the hub management status field is directly appended to the received data at a byte boundary. If the repeated packet is significantly longer than the value in the Decode Register requires and PCOMP is asserted the status fields will be delayed until the end of packet repetition. During this delay period MRXC clocks are inhibited but the MCRS signal remains asserted.

**Note:** If PCOMP is asserted late in the packet, i.e., after the number of bytes defined by the packet compression register, then packet compression will not occur.

The Management Interface may be fine tuned to meet the timing consideration of the SONIC and the access time of its associated packet memory. This refinement may be performed in two ways:

1. The default mode of operation of the Management interface is to only transfer packets over the bus which have a start of frame delimiter. Thus "packets" that are only preamble/jam and do not convey any source or destination address information are inhibited. This filtering may be disabled by writing a logic zero to the Management Interface Configuration or "MIFCON" bit in the Management and Interrupt Configuration Register. See Section 8 for details.
2. The Management bus has been designed to accommodate situations of maximum network utilization, for example when collision generated fragments occur; (these collision fragments may violate the IEEE802.3 IFG specification). The IFG required by the SONIC is a function of the time taken to release space in the receive FIFO and to perform end of packet processing (write status information into memory). These functions are primarily memory operations and consequently depend upon the bus latency and the memory access time of the system. In order to allow the system designer some discretion in choosing the speed of this memory, the RIC may be configured to protect the SONIC from a potential FIFO overflow. This is performed by utilizing the Inter Frame Gap Threshold Select Register.

The value held in this register, plus one, defines, in network bit times, the minimum allowed gap between frames on the management bus. If the gap is smaller than this number then MCRS is asserted but MRXC clocks are inhibited. Consequently no data transfer is performed.

Thus the system designer may make the decision whether to gather statistics on all packets even if they occur with very small IFGs or to monitor a subset.

The status field, shown in *Figure 6.1*, contains information which may be conveniently analyzed by considering it as

## 6.0 Hub Management Support (Continued)

providing information of six different types. They are held in seven Packet Status Registers "PSRs":

1. The RIC and port address fields [PSR(0) and (1)] can uniquely identify the repeater port receiving the packet out of a potential maximum of 832 ports sharing the same management bus (64 RICs each with 13 ports). Thus all of the other status fields can be correctly attributed to the relevant port.
2. The status flags the RIC produces for the event counters or recording latches are supplied with each packet [PSR(2)]. Additionally the clean receive CLN status is supplied to allow the user to determine the reliability of the address fields in the packet. The CLN status bit [PSR(1)] is set if no collisions are experienced during the repetition of the address fields.
3. The RIC has an on-chip timer to indicate when, relative to the start of packet repetition, a collision, if any, occurred [PSR(3)]. There is also a timer which indicates how many bit times of IFG was seen on the network between repetition of this packet and the preceding one. This is provided by [PSR(6)].
4. If packet compression is employed, the receive byte count contained in the SONIC's packet descriptor will indicate the number of bytes transferred over the management bus rather than the number of bytes in the packet. For this reason the RIC which receives the packet,

counts the number of received bytes and transfers this over the management bus [PSR(4), (5)].

5. Appending a status field to a data packet will obviously result in a CRC error being flagged by the SONIC. For this reason the RIC monitors the repeated data stream to check for CRC and FAE errors. In the case of FAE errors the RIC provides additional dummy data bits, so that the status fields are always byte aligned.
6. As a final check upon the effectiveness of the management interface, the RIC transfers a bus specific status bit to the SONIC. This flag Packet Compress Done PCOMP [PSR(0)], may be monitored by hub management software to check if the packet compression operation is enabled.

*Figure 6.2* shows an example of a packet being transmitted over the management bus. The first section of the diagram (moving from left to right) shows a short preamble and SFD pattern. The second region contains the packet's address and the start of the data fields. During this time logic on the processor/SONIC card would determine if packet compression should be used on this packet. The PCOMP signal is asserted and packet transfer stops when the number of bytes transmitted equals the value defined in the decode register. Hence the MRXC signal is idle for the remainder of the packet's data and CRC fields. The final region shows the transfer of the RIC's seven bytes of packet status.

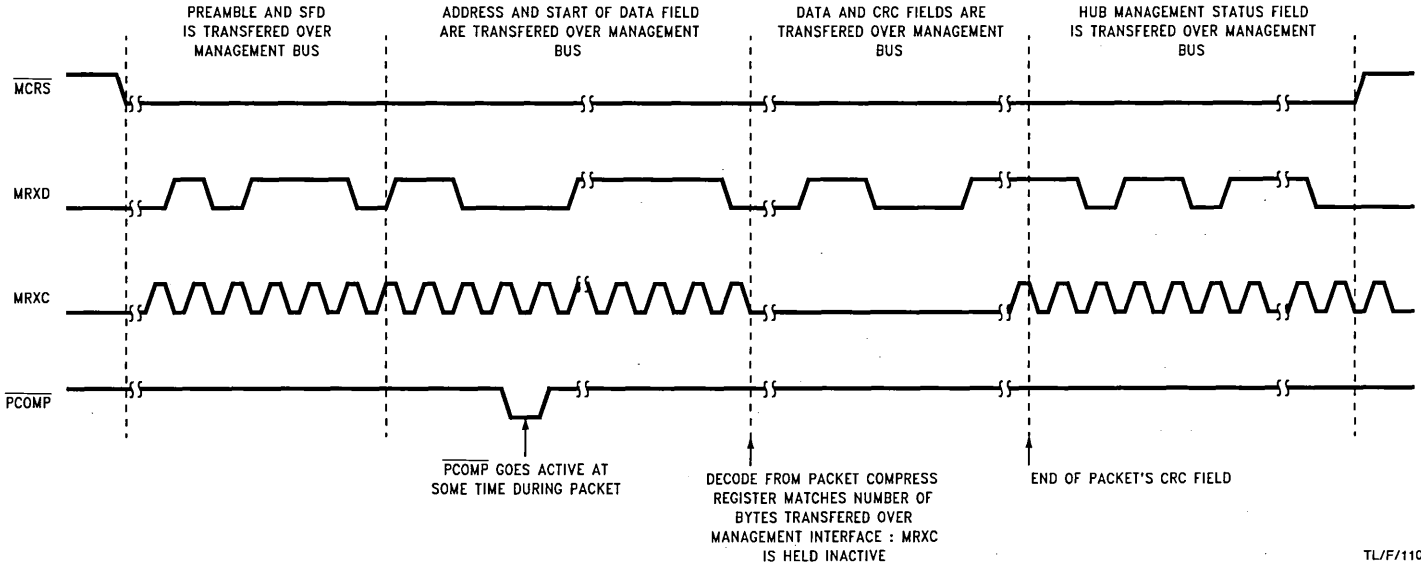
The following pages describe these Hub Management registers which constitute the management status field.

| Packet Status Register PSR             | D7    | D6    | D5    | D4    | D3    | D2    | D1    | D0    |
|----------------------------------------|-------|-------|-------|-------|-------|-------|-------|-------|
| PSR(0)                                 | A5    | A4    | A3    | A2    | A1    | A0    | PCOMP | TXCOL |
| PSR(1)                                 | CRCER | FAE   | COL   | CLN   | PA3   | PA2   | PA1   | PA0   |
| PSR(2)                                 | SE    | OWC   | NSFD  | PLER  | ELBER | JAB   | CBT9  | CBT8  |
| PSR(3)<br>Collision Bit<br>Timer       | CBT7  | CBT6  | CBT5  | CBT4  | CBT3  | CBT2  | CBT1  | CBT0  |
| PSR(4)<br>Lower Repeat<br>Byte Count   | RBY7  | RBY6  | RBY5  | RBY4  | RBY3  | RBY2  | RBY1  | RBY0  |
| PSR(5)<br>Upper Repeat<br>Byte Count   | RBY15 | RBY14 | RBY13 | RBY12 | RBY11 | RBY10 | RBY9  | RBY8  |
| PSR(6)<br>Inter Frame<br>Gap Bit Timer | IBT7  | IBT6  | IBT5  | IBT4  | IBT3  | IBT2  | IBT1  | IBT0  |

**Note:** These registers may only be reliably accessed via the management interface. Due to the nature of these registers they may not be accessed (read or write cycles) via the processor interface.

**FIGURE 6.1. Hub Management Status Field**





Note: In this example the Management Bus is configured to use active low signals.

FIGURE 6.2. Operation of the Management Bus

TL/F/11096-18

3-49

## 6.0 Hub Management Support (Continued)

### Packet Status Register 0

|    |    |    |    |    |    |       |      |
|----|----|----|----|----|----|-------|------|
| D7 | D6 | D5 | D4 | D3 | D2 | D1    | D0   |
| A5 | A4 | A3 | A2 | A1 | A0 | PCOMP | resv |

| Bit    | Symbol | Description                                                                                                                                                                                                      |
|--------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0     | resv   | <b>RESERVED FOR FUTURE USE:</b> This bit is currently undefined, management software should not examine the state of this bit.                                                                                   |
| D1     | PCOMP  | <b>PACKET COMPRESSION DONE:</b> If packet compression is utilized, this bit informs the user that compression was performed, i.e., the packet was long enough to require compression.                            |
| D(7:2) | A(5:0) | <b>RIC ADDRESS (5:0):</b> This address is defined by the user and is supplied when writing to the RIC Address Register. It is used by hub management software to distinguish between RICs in a multi-RIC system. |

### Packet Status Register 1

|       |     |     |     |     |     |     |     |
|-------|-----|-----|-----|-----|-----|-----|-----|
| D7    | D6  | D5  | D4  | D3  | D2  | D1  | D0  |
| CRCER | FAE | COL | CLN | PA3 | PA2 | PA1 | PA0 |

| Bit    | Symbol  | Description                                                                                                                                                                                                                                                                        |
|--------|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D(3:0) | PA(3:0) | <b>PORT ADDRESS:</b> This field defines the port which is receiving the packet.                                                                                                                                                                                                    |
| D4     | CLN     | <b>CLEAN RECEIVE:</b> This bit is asserted from the start of reception, and is deasserted if a collision occurs within a window from the start of reception to the end of the 13th byte after SFD detection. If no SFD is detected the window is extended to the end of reception. |
| D5     | COL     | <b>COLLISION:</b> If a receive or transmit collision occurs during packet repetition the collision bit is asserted.                                                                                                                                                                |
| D6     | FAE     | <b>FRAME ALIGNMENT ERROR:</b> This bit is asserted if a Frame Alignment Error occurred in the repeated packet.                                                                                                                                                                     |
| D7     | CRCER   | <b>CRC ERROR:</b> This bit is asserted if a CRC Error occurred in the repeated packet. This status flag should not be tested if the COL bit is asserted since the error may be simply due to the collision.                                                                        |

## 6.0 Hub Management Support (Continued)

### Packet Status Register 2

| D7 | D6  | D5   | D4   | D3    | D2  | D1   | D0   |
|----|-----|------|------|-------|-----|------|------|
| SE | OWC | NSFD | PLER | ELBER | JAB | CBT9 | CBT8 |

| Bit    | Symbol  | Description                                                                                                                                                                                                                                                                                                                                                                                                         |
|--------|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D(1:0) | CT(9:8) | <b>COLLISION TIMER BITS 9 AND 8:</b> These two bits are the upper bits of the collision bit timer.                                                                                                                                                                                                                                                                                                                  |
| D2     | JAB     | <b>JABBER EVENT:</b> This bit indicates that the receive packet was so long the repeater was forced to go into a jabber protect condition.                                                                                                                                                                                                                                                                          |
| D3     | ELBER   | <b>ELASTICITY BUFFER ERROR:</b> During the packet an Elasticity Buffer under/overflow occurred.                                                                                                                                                                                                                                                                                                                     |
| D4     | PLER    | <b>PHASE LOCK LOOP ERROR:</b> The packet suffered sufficient jitter/noise corruption to cause the phase lock loop decoder to lose lock.                                                                                                                                                                                                                                                                             |
| D5     | NSFD    | <b>NON SFD:</b> The repeated packet did not contain a Start of Frame Delimiter. When this bit is set the Repeat Byte Counter counts the length of the entire packet. When this bit is not set the byte counter only counts post SFD bytes.<br><b>Note:</b> The operation of this bit is not inhibited by the occurrence of a collision during packet repetition (see description of the Repeat Byte Counter below). |
| D6     | OWC     | <b>OUT OF WINDOW COLLISION:</b> The packet suffered an out of window collision.                                                                                                                                                                                                                                                                                                                                     |
| D7     | SE      | <b>SHORT EVENT:</b> The receive activity was so small it met the criteria to be classed as a short event.                                                                                                                                                                                                                                                                                                           |

The other registers comprise the remainder of the collision timer register [PSR(3)], the Repeat Byte Count registers [PSR(4), (5)], and the Inter Frame Gap Counter "IFG" register [PSR(6)].

#### Collision Bit Timer

The Collision Timer counts in bit times the time between the start of repetition of the packet and the detection of the packet's first collision. The Collision counter increments as the packet is repeated and freezes when a collision occurs. The value in the counter is only valid when the collision bit "COL" in [PSR(1)] is set.

#### Repeat Byte Counter

The Repeat Byte Counter is a 16 bit counter which can perform two functions. In cases where the transmitted packet possesses an SFD, the byte counter counts the number of received bytes after the SFD field. Alternatively if no SFD is repeated the counter reflects the length of the packet, counted in bytes, starting at the beginning of the preamble field. When performing the latter function the counter is shortened to 7 bits. Thus the maximum count value is 127 bytes. The mode of counting is indicated by the "NSFD" bit in [PSR(2)]. In order to check if the received packet was genuinely a Non-SFD packet, the status of the COL bit should be checked. During collisions SFD fields may be lost or created, Management software should be robust to this kind of behaviour.

#### Inter Frame Gap (IFG) Bit Timer

The IFG counter counts in bit times the period in between repeater transmissions. The IFG counter increments whenever the RIC is not transmitting a packet. If the IFG is long, i.e., greater than 255 bits the counter sticks at this value. Thus an apparent count value of 255 should be interpreted as 255 or more bit times.

#### 6.4 DESCRIPTION OF HARDWARE CONNECTION FOR MANAGEMENT INTERFACE

The RIC has been designed so it may be connected to the Management bus directly or via external bus transceivers. The latter is advantageous in large repeaters. In this application the system backplane is often heavily loaded beyond the drive capabilities of the on-chip bus drivers.

The uni-directional nature of information transfer on the MCRS, MRXD and MRXC signals, means a single open drain output pin is adequate for each of these signals. The Management Enable (MEN) RIC output pin performs the function of a drive enable for an external bus transceiver if one is required.

In common with the Inter-RIC bus signals ACTN, ANYXN, COLN and IRE the MCRS active level asserted by the MCRS output is determined by the state of the BINV Mode Load configuration bit.

#### 7.0 Port Block Functions

The RIC has 13 port logic blocks (one for each network connection). In addition to the packet repetition operations already described, the port block performs two other functions:

1. The physical connection to the network segment (transceiver function).
2. It provides a means to protect the network from malfunctioning segments (segment partition).

Each port has its own status register. This register allows the user to determine the current status of the port and configure a number of port specific functions.

## 7.0 Port Block Functions (Continued)

### 7.1 TRANSCEIVER FUNCTIONS

The RIC may connect to network segments in three ways:

1. Over AUI cable to transceiver boxes,
2. Directly to board mounted transceivers,
3. To twisted pair cable via a simple interface.

The first method is only supported by RIC port 1 (the AUI port). Options (2) and (3) are available on ports 2 to 13. The selection of the desired option is made at device initialization during the Mode Load operation. The Transceiver Bypass XBYPAS configuration bits are used to determine whether the ports will utilize the on-chip 10BASE-T transceiver or bypass these in favour of external transceivers. Four possible combinations of port utilization are supported: All ports (2 to 13) use the external Transceiver Interface.

Ports 2 to 5 use the external interface, 6 to 13 use the internal 10BASE-T transceivers.

Ports 2 to 7 use the external interface, 8 to 13 use the internal 10BASE-T transceivers.

All ports (2 to 13) use the internal 10BASE-T transceivers.

#### 10BASE-T Transceiver Operation

The RIC contains virtually all the digital and analog circuits required for connection to 10BASE-T network segments. The only additional active component is an external driver packet. The connection for a RIC port to a 10BASE-T segment is shown in *Figure 7.1*. The diagram shows the components required to connect one of the RIC's ports to a 10BASE-T segment. The major components are the driver package, a member of the 74ACT family, and an integrated filter/choke network.

The operation of the 10BASE-T transceiver's logical functions may be modified by software control. The default mode of operation is for the transceivers to transmit and expect reception of link pulses. This may be modified if a logic one is written to the  $\overline{\text{GDLNK}}$  bit of a port's status register. The port's transceiver will operate normally but will not transmit link pulses nor monitor their reception. Thus the entry to a link fail state and the associated modification of transceiver operation will not occur.

The on-chip 10BASE-T transceivers automatically detect and correct the polarity of the received data stream. This polarity detection scheme relies upon the polarity of the received link pulses and the end of the packet waveform. Polarity detection and correction may be disabled under software control as follows:

- 1) Write the value 07H to the Page Select Register (address 10H).
- 2) Write the value 02H to the address 11H. (Note that address 11H will read back 00H after writing 02H to it).

This is the only exception for accessing any of the reserved pages 4 to 7.

#### External Transceiver Operation

RIC ports 2 to 13 may be connected to media other than twisted-pair by opting to bypass the on-chip transceivers. When using external transceivers the user must have the external transceivers perform collision detection and the other functions associated with an IEEE 802.2 Media Access Unit. *Figure 7.2* shows the connection between a repeater port and a coaxial transceiver using the AUI type interface.

### 7.2 SEGMENT PARTITION

Each of the RIC's ports has a dedicated state machine to perform the functions defined by the IEEE partition algorithm as shown in *Figure 7.3*. To allow users to customize this algorithm for different applications a number of user selected options are available during device configuration at power up (the Mode Load Cycle).

Five different options are provided:

1. Operation of the 13 partition state machines may be disabled via the disable partition  $\overline{\text{DPART}}$  configuration bit (Pin D6).
2. The value of consecutive counts required to partition a segment (the  $\text{CCLimit}$  specification) may be set at either 31 or 63 consecutive collisions.
3. The use of the TW5 specification in the partition algorithm differentiates between collisions which occur early in a packet (before TW5 has elapsed) and those which occur late in the packet (after TW5 has elapsed). These late or "out of window" collisions can be regarded in the same manner as early collisions if the Out of Window Collision Enable  $\overline{\text{OWCE}}$  option is selected. This configuration bit is applied to the D4 pin during the Mode Load operation. The use of  $\overline{\text{OWCE}}$  delays until the end of the packet the operation of the state diagram branch marked (1) and enables the branch marked (2) in *Figure 7.3*.
4. The operation of the ports' state machines when reconnecting a segment may also be modified by the user. The Transmit Only  $\overline{\text{TXONLY}}$  configuration bit allows the user to prevent segment reconnection unless the reconnecting packet is being sourced by the repeater. In this case the repeater is transmitting on to the segment, rather than the segment transmitting when the repeater is idle. The normal mode of reconnection does not differentiate between such packets. The  $\overline{\text{TXONLY}}$  configuration bit is input on Pin D5 during the Mode Load cycle. If this option is selected the operation of the state machine branch marked (3) in *Figure 7.3* is affected.
5. The RIC may be configured to use an additional criterion for segment partition. This is referred to as loop back partition. If this operation is selected the partition state machine monitors the receive and collision inputs from a network segment to discover if they are active when the port is transmitting. Thus determining if the network transceiver is looping back the data pattern from the cable. A port may be partitioned if no data or collision signals are seen by the partition logic in the following window: 61 to 96 network bit times after the start of transmission see data sheet Section 8 for details. A segment partitioned by this operation may be reconnected in the normal manner.

In addition to the autonomous operation of the partition state machines, the user may reset these state machines. This may be done individually to each port by writing a logic one to the  $\overline{\text{PART}}$  bit in its status register. The port's partition state machine and associated counters are reset and the port is reconnected to the network. The reason why a port become partitioned may be discovered by the user by reading the port's status register.

## 7.0 Port Block Functions (Continued)

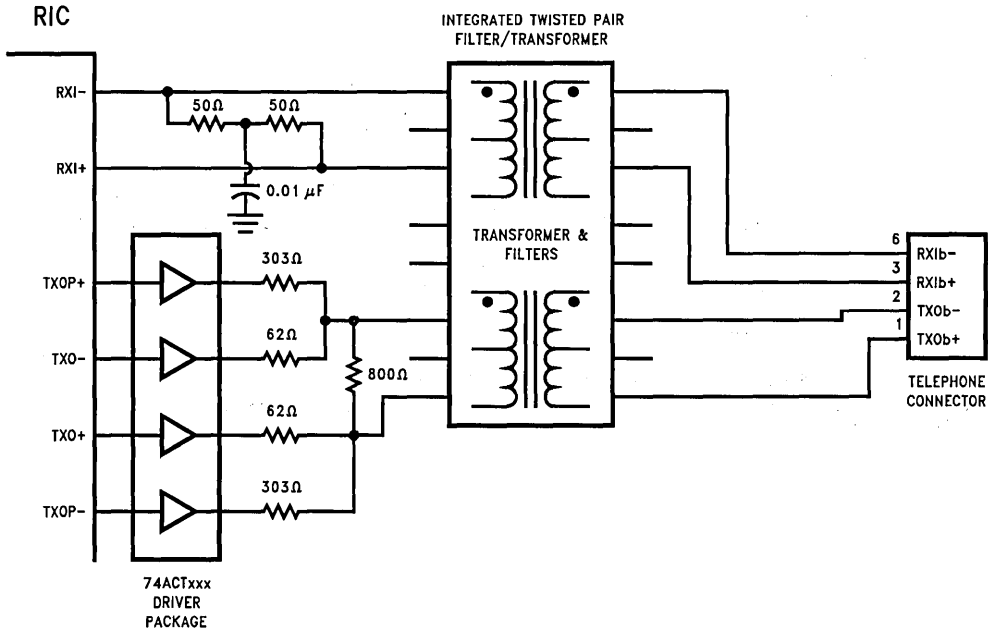
### 7.3 PORT STATUS REGISTER FUNCTIONS

Each RIC port has its own status register. In addition to providing status concerning the port and its network segment the register allows the following operations to be performed upon the port:

1. Port disable
2. Link Disable
3. Partition reconnection
4. Selection between normal and reduced squelch levels

Note that the link disable and port disable functions are mutually exclusive functions, i.e., disabling link does not affect receiving and transmitting from/to that port and disabling a port does not disable link.

When a port is disabled packet transmission and reception between the port's segment and the rest of the network is prevented.

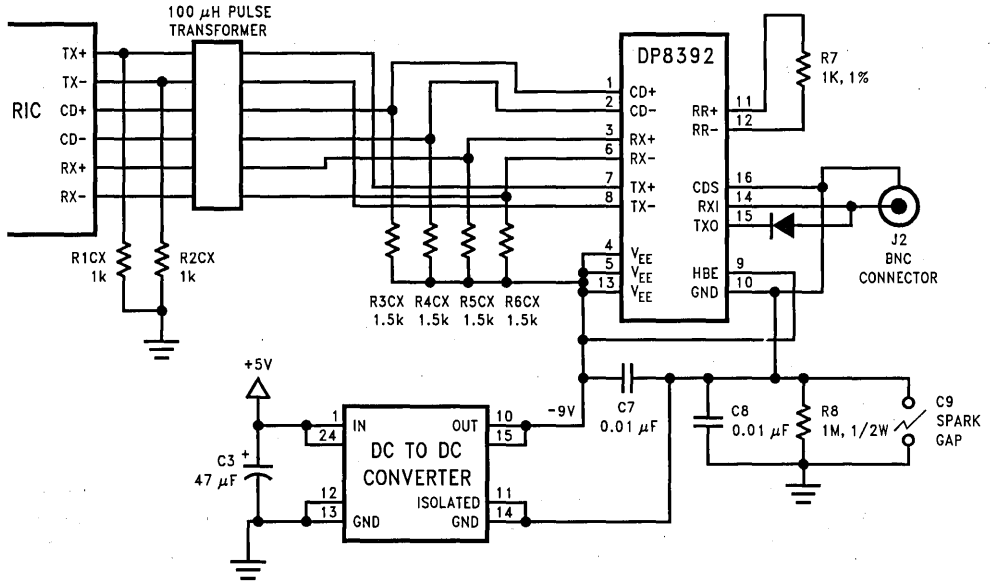


Note: For recommended modules, see "Ethernet Magnetics Vendors for 10BASE-T, 10BASE2, and 10BASE5" in Section 5 of this Databook.

TL/F/11096-19

FIGURE 7.1. Port Connection to a 10BASE-T Segment

### 7.0 Port Block Functions (Continued)



TL/F/11096-20

**FIGURE 7.2. Port Connection to a 10BASE2 Segment (AUI Interface Selected)**

The preceding diagrams show a RIC port (Numbers 2 to 13) connected to a 10BASE-T and a 10BASE2 segment. The values of any components not indicated above are to be determined.

7.0 Port Block Functions (Continued)

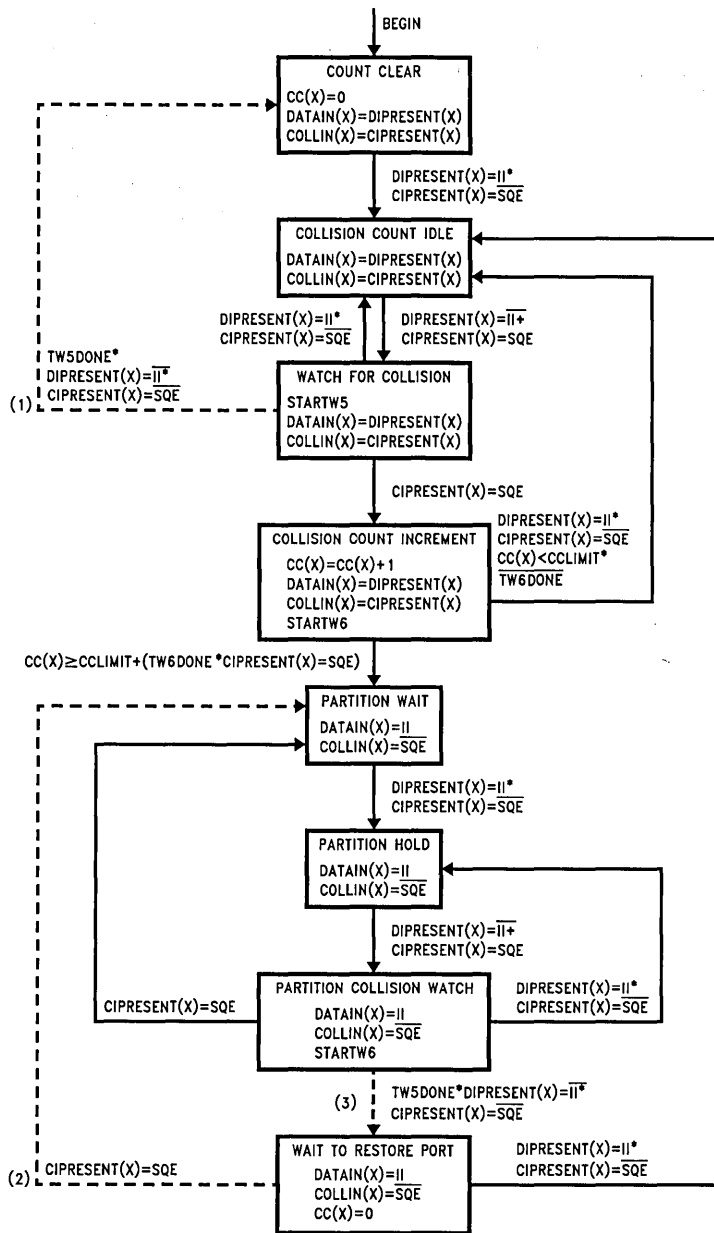


FIGURE 7.3. IEEE Segment Partition Algorithm

TL/F/11096-21

## 8.0 RIC Registers

### RIC Register Address Map

The RIC's registers may be accessed by applying the required address to the five Register Address (RA(4:0)) input pins. Pin RA4 makes the selection between the upper and lower halves of the register array. The lower half of the register map consists of 16 registers:

- 1 RIC Real Time Status and Configuration register,
- 13 Port Real Time Status registers,
- 1 RIC Configuration Register
- 1 Real Time Interrupt Status Register.

These registers may be directly accessed at any time via the RA(4:0) pins, (RA4 = 0). The upper half of the register map, (RA4 = 1), is organized as 4 pages of registers:

- Event Count Configuration page (0),
- Event Record page (1),
- Lower Event Count page (2)
- Upper Event Count page (3)

Register access within these pages is also performed using the RA(4:0) pins, (RA4 = 1). Page switching is performed by writing to the Page Selection bits (PSEL2, 1, 0). These bits are found in the Page Select Register, located at address 10 hex on each page of the upper half of the register array. AT power on these bits default to 0 Hex, i.e., page zero.



## 8.0 RIC Registers (Continued)

Register Memory Map

| Address | Name                                        |                                    |                                         |                   |
|---------|---------------------------------------------|------------------------------------|-----------------------------------------|-------------------|
|         | Page (0)                                    | Page (1)                           | Page (2)                                | Page (3)          |
| 00H     | RIC Status and Configuration Register       |                                    |                                         |                   |
| 01H     | Port 1 Status Register                      |                                    |                                         |                   |
| 02H     | Port 2 Status Register                      |                                    |                                         |                   |
| 03H     | Port 3 Status Register                      |                                    |                                         |                   |
| 04H     | Port 4 Status Register                      |                                    |                                         |                   |
| 05H     | Port 5 Status Register                      |                                    |                                         |                   |
| 06H     | Port 6 Status Register                      |                                    |                                         |                   |
| 07H     | Port 7 Status Register                      |                                    |                                         |                   |
| 08H     | Port 8 Status Register                      |                                    |                                         |                   |
| 09H     | Port 9 Status Register                      |                                    |                                         |                   |
| 0AH     | Port 10 Status Register                     |                                    |                                         |                   |
| 0BH     | Port 11 Status Register                     |                                    |                                         |                   |
| 0CH     | Port 12 Status Register                     |                                    |                                         |                   |
| 0DH     | Port 13 Status Register                     |                                    |                                         |                   |
| 0EH     | RIC Configuration Register                  |                                    |                                         |                   |
| 0FH     | Real Time Interrupt Register                |                                    |                                         |                   |
| 10H     | Page Select Register                        |                                    |                                         |                   |
| 11H     | Device Type Register                        | Port 1 Event Record Register (ERR) |                                         |                   |
| 12H     | Lower Event Count Mask Register (ECMR)      | Port 2 ERR                         | Port 1 Lower Event Count Register (ECR) | Port 8 Lower ECR  |
| 13H     | Upper ECMR                                  | Port 3 ERR                         | Port 1 Upper ECR                        | Port 8 Upper ECR  |
| 14H     | Event Record Mask Register                  | Port 4 ERR                         | Port 2 Lower ECR                        | Port 9 Lower ECR  |
| 15H     | resv                                        | Port 5 ERR                         | Port 2 Upper ECR                        | Port 9 Upper ECR  |
| 16H     | Management/Interrupt Configuration Register | Port 6 ERR                         | Port 3 Lower ECR                        | Port 10 Lower ECR |
| 17H     | RIC Address Register                        | Port 7 ERR                         | Port 3 Upper ECR                        | Port 10 Upper ECR |
| 18H     | Packet Compress Decode Register             | Port 8 ERR                         | Port 4 Lower ECR                        | Port 11 Lower ECR |
| 19H     | resv                                        | Port 9 ERR                         | Port 4 Upper ECR                        | Port 11 Upper ECR |
| 1AH     | resv                                        | Port 10 ERR                        | Port 5 Lower ECR                        | Port 12 Lower ECR |
| 1BH     | resv                                        | Port 11 ERR                        | Port 5 Upper ECR                        | Port 12 Upper ECR |
| 1CH     | resv                                        | Port 12 ERR                        | Port 6 Lower ECR                        | Port 13 Lower ECR |
| 1DH     | resv                                        | Port 13 ERR                        | Port 6 Upper ECR                        | Port 13 Upper ECR |
| 1EH     | resv                                        |                                    | Port 7 Lower ECR                        |                   |
| 1FH     | IFG Threshold                               |                                    | Port 7 Upper ECR                        |                   |

Note: All registers marked resv on pages 0 to 3 must not be accessed by the user. The other register pages, 4 to 7, are also reserved.

## 8.0 RIC Registers (Continued)

### Register Array Bit Map Addresses 00H to 10H

| Address (Hex) | D7     | D6     | D5     | D4     | D3     | D2    | D1    | D0    |
|---------------|--------|--------|--------|--------|--------|-------|-------|-------|
| 00            | BINV   | BYPAS2 | BYPAS1 | APART  | JAB    | AREC  | ACOL  | resv  |
| 01 to 0D      | DISPT  | SQL    | PTYPE1 | PTYPE0 | PART   | REC   | COL   | GDLNK |
| 0E            | MINMAX | DPART  | TXONLY | OWCE   | LPPART | CCLIM | Tw2   | resv  |
| 0F            | IVCTR3 | IVCTR2 | IVCTR1 | IVCTR0 | ISRC3  | ISRC2 | ISRC1 | ISRC0 |

### Register Array Bit Map Addresses 10H to 1FH Page (0)

| Address (Hex) | D7     | D6    | D5    | D4     | D3     | D2    | D1     | D0     |
|---------------|--------|-------|-------|--------|--------|-------|--------|--------|
| 10            | FC     | HC    | LC    | FF     | resv   | PSEL2 | PSEL1  | PSEL0  |
| 11            | 0      | 0     | 0     | 0      | 0      | 0     | 0      | 0      |
| 12            | BDLNK  | PARTC | RECC  | SEC    | NSFDC  | PLERC | ELBERC | JABC   |
| 13            | resv   | resv  | OWCC  | RXCOLC | TXCOLC | resv  | FWF    | ROR    |
| 14            | BDLNKE | PARTE | OWCE  | SEE    | NSFDE  | PLERE | ELBERE | JABE   |
| 16            | IFC    | IHC   | ILC   | IFF    | IREC   | ICOL  | IPART  | MIFCON |
| 17            | A5     | A4    | A3    | A2     | A1     | A0    | resv   | resv   |
| 18            | PCD7   | PCD6  | PCD5  | PCD4   | PCD3   | PCD2  | PCD1   | PCD0   |
| 1F            | IFGT7  | IFGT6 | IFGT5 | IFGT4  | IFGT3  | IFGT2 | IFGT1  | IFGT0  |

### Register Array Bit Map Addresses 10H to 1FH Page (1)

| Address (Hex) | D7    | D6   | D5  | D4 | D3   | D2    | D1    | D0    |
|---------------|-------|------|-----|----|------|-------|-------|-------|
| 10            | FC    | HC   | LC  | FF | resv | PSEL2 | PSEL1 | PSEL0 |
| 11 to 1D      | BDLNK | PART | OWC | SE | NSFD | PLER  | ELBER | JAB   |

### Register Array Bit Map Addresses 10H to 1FH Pages (2) and (3)

| Address (Hex)  | D7   | D6   | D5   | D4   | D3   | D2    | D1    | D0    |
|----------------|------|------|------|------|------|-------|-------|-------|
| 10             | FC   | HC   | LC   | FF   | resv | PSEL2 | PSEL1 | PSEL0 |
| 11             | —    | —    | —    | —    | —    | —     | —     | —     |
| Even Locations | EC7  | EC6  | EC5  | EC4  | EC3  | EC2   | EC1   | EC0   |
| Odd Locations  | EC15 | EC14 | EC13 | EC12 | EC11 | EC10  | EC9   | EC8   |

## 8.0 RIC Registers (Continued)

### RIC Status and Configuration Register (Address 00H)

The lower portion of this register contains real time information concerning the operation of the RIC. The upper three bits represent the chosen configuration of the transceiver interface employed.

| D7   | D6     | D5     | D4    | D3  | D2   | D1   | D0   |
|------|--------|--------|-------|-----|------|------|------|
| BINV | BYPAS2 | BYPAS1 | APART | JAB | AREC | ACOL | resv |

| Bit | R/W | Symbol Access | Description                                                                                                                                                                                                   |
|-----|-----|---------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0  | R   | resv          | <b>RESERVED FOR FUTURE USE:</b><br>Reads as a logic 0.                                                                                                                                                        |
| D1  | R   | ACOL          | <b>ANY COLLISIONS:</b><br>0: A collision is occurring at one or more of the RIC's ports.<br>1: No collisions.                                                                                                 |
| D2  | R   | AREC          | <b>ANY RECEIVE:</b><br>0: One of the RIC's ports is the current packet or collision receiver.<br>1: No packet or collision reception within this RIC.                                                         |
| D3  | R   | JAB           | <b>JABBER PROTECT:</b><br>0: The RIC has been forced into jabber protect state by one of its ports or by another port on the Inter-RIC bus, (Multi-RIC operations).<br>1: No jabber protect conditions exist. |
| D4  | R   | APART         | <b>ANY PARTITION:</b><br>0: One or more ports are partitioned.<br>1: No ports are partitioned.                                                                                                                |
| D5  | R   | BYPAS1        | These bits define the configuration of ports 2 to 13 i.e., their use if the internal 10BASE-T transceivers or the external (AUI-like) transceiver interface.                                                  |
| D6  | R   | BYPAS2        |                                                                                                                                                                                                               |
| D7  | R   | BINV          | <b>BUS INVERT:</b><br>This register bit informs whether the Inter-RIC signals: IRE, ACTN, ANYXN, COLN and Management bus signal MCRS are active high or low.<br>0: Active high<br>1: Active low               |

## 8.0 RIC Registers (Continued)

Port Real Time Status Registers (Address 01H to 0DH)

|       |     |        |        |      |     |     |       |
|-------|-----|--------|--------|------|-----|-----|-------|
| D7    | D6  | D5     | D4     | D3   | D2  | D1  | D0    |
| DISPT | EGP | PTYPE1 | PTYPE0 | PART | REC | COL | GDLNK |

| Bit     | R/W    | Symbol                                                            | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |        |        |             |   |   |                                     |   |   |                                             |   |   |                                                                   |   |   |                               |
|---------|--------|-------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|--------|-------------|---|---|-------------------------------------|---|---|---------------------------------------------|---|---|-------------------------------------------------------------------|---|---|-------------------------------|
| D0      | R/W    | GDLNK                                                             | <p><b>GOOD LINK:</b><br/>                     0: Link pulses are being received by the port.<br/>                     1: Link pulses are not being received by the port logic.<br/> <b>Note:</b> Writing a 1 to this bit will cause the 10BASE-T transceiver not to transmit or monitor the reception of link pulses. If the internal 10BASE-T transceivers are not selected or if port 1 (AUI port) is read, then this bit is undefined.</p>                                                                                                                                                                                                                                             |        |        |             |   |   |                                     |   |   |                                             |   |   |                                                                   |   |   |                               |
| D1      | R      | COL                                                               | <p><b>COLLISION:</b><br/>                     0: A collision is happening or has occurred during the current packet.<br/>                     1: No collisions have occurred as yet during this packet.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |        |        |             |   |   |                                     |   |   |                                             |   |   |                                                                   |   |   |                               |
| D2      | R      | REC                                                               | <p><b>RECEIVE:</b><br/>                     0: This port is now or has been the receive source of packet or collision information for the current packet.<br/>                     1: This port has not been the receive source during the current packet.</p>                                                                                                                                                                                                                                                                                                                                                                                                                            |        |        |             |   |   |                                     |   |   |                                             |   |   |                                                                   |   |   |                               |
| D3      | R/W    | PART                                                              | <p><b>PARTITION:</b><br/>                     0: This port is partitioned.<br/>                     1: This port is not partitioned.<br/>                     Writing a logic one to this bit forces segment reconnection and partition state machine reset.<br/>                     Writing a zero to this bit has no effect.</p>                                                                                                                                                                                                                                                                                                                                                       |        |        |             |   |   |                                     |   |   |                                             |   |   |                                                                   |   |   |                               |
| D(5, 4) | R      | PTYPE0<br>PTYPE1                                                  | <p><b>PARTITION TYPE 0</b><br/> <b>PARTITION TYPE 1</b><br/>                     The partition type bits provide information specifying why the port is partitioned.</p> <table border="1" style="margin-left: 20px;"> <thead> <tr> <th>PTYPE1</th> <th>PTYPE0</th> <th>Information</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Consecutive Collision Limit Reached</td> </tr> <tr> <td>0</td> <td>1</td> <td>Excessive Length of Collision Limit Reached</td> </tr> <tr> <td>1</td> <td>0</td> <td>Failure to See Data Loopback from Transceiver in Monitored Window</td> </tr> <tr> <td>1</td> <td>1</td> <td>Processor Forced Reconnection</td> </tr> </tbody> </table> | PTYPE1 | PTYPE0 | Information | 0 | 0 | Consecutive Collision Limit Reached | 0 | 1 | Excessive Length of Collision Limit Reached | 1 | 0 | Failure to See Data Loopback from Transceiver in Monitored Window | 1 | 1 | Processor Forced Reconnection |
| PTYPE1  | PTYPE0 | Information                                                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |        |        |             |   |   |                                     |   |   |                                             |   |   |                                                                   |   |   |                               |
| 0       | 0      | Consecutive Collision Limit Reached                               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |        |        |             |   |   |                                     |   |   |                                             |   |   |                                                                   |   |   |                               |
| 0       | 1      | Excessive Length of Collision Limit Reached                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |        |        |             |   |   |                                     |   |   |                                             |   |   |                                                                   |   |   |                               |
| 1       | 0      | Failure to See Data Loopback from Transceiver in Monitored Window |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |        |        |             |   |   |                                     |   |   |                                             |   |   |                                                                   |   |   |                               |
| 1       | 1      | Processor Forced Reconnection                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |        |        |             |   |   |                                     |   |   |                                             |   |   |                                                                   |   |   |                               |
| D6      | R/W    | SQL                                                               | <p><b>SQUELCH LEVEL:</b><br/>                     0: Port operates with normal IEEE receive squelch level.<br/>                     1: Port operates with reduced receive squelch level.<br/> <b>Note:</b> This bit has no effect when the external transceiver is selected.</p>                                                                                                                                                                                                                                                                                                                                                                                                          |        |        |             |   |   |                                     |   |   |                                             |   |   |                                                                   |   |   |                               |
| D7      | R/W    | DISPT                                                             | <p><b>DISABLE PORT:</b><br/>                     0: Port operates as defined by repeater operations.<br/>                     1: All port activity is prevented.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |        |        |             |   |   |                                     |   |   |                                             |   |   |                                                                   |   |   |                               |

## 8.0 RIC Registers (Continued)

### RIC Configuration Register (Address 0EH)

This register displays the state of a number of RIC configuration bits loaded during the Mode Load operation.

| D7     | D6    | D5     | D4   | D3     | D2    | D1  | D0   |
|--------|-------|--------|------|--------|-------|-----|------|
| MINMAX | DPART | TXONLY | OWCE | LPPART | CCLIM | Tw2 | resv |

| Bit | R/W | Symbol | Description                                                                                                                                                                                                                                                                                                                                                                           |
|-----|-----|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0  | R   | resv   | <b>RESERVED FOR FUTURE USE:</b> Value set at logic one.                                                                                                                                                                                                                                                                                                                               |
| D1  | R   | Tw2    | <b>CARRIER RECOVERY TIME:</b><br>0: Tw2 set at 5 bits.<br>1: Tw2 set at 3 bits.                                                                                                                                                                                                                                                                                                       |
| D2  | R   | CCLIM  | <b>CONSECUTIVE COLLISION LIMIT:</b><br>0: Consecutive collision limit set at 63 collisions.<br>1: Consecutive collision limit set at 31 collisions.                                                                                                                                                                                                                                   |
| D3  | R   | LPPART | <b>LOOPBACK PARTITION:</b><br>0: Partitioning upon lack of loopback from transceivers is enabled.<br>1: Partitioning upon lack of loopback from transceivers is disabled.                                                                                                                                                                                                             |
| D4  | R   | OWCE   | <b>OUT OF WINDOW COLLISION ENABLE:</b><br>0: Out of window collisions are treated as in window collisions by the segment partition state machines.<br>1: Out of window collisions are treated as out of window collisions by the segment partition state machines.                                                                                                                    |
| D5  | R   | TXONLY | <b>ONLY RECONNECT UPON SEGMENT TRANSMISSION:</b><br>0: A segment will only be reconnected to the network if a packet transmitted by the RIC onto that segment fulfills the requirements of the segment reconnection algorithm.<br>1: A segment will be reconnected to the network by any packet on the network which fulfills the requirements of the segment reconnection algorithm. |
| D6  | R   | DPART  | <b>DISABLE PARTITION:</b><br>0: Partitioning of ports by on-chip algorithms is prevented.<br>1: Partitioning of ports by on-chip algorithms is enabled.                                                                                                                                                                                                                               |
| D7  | R   | MINMAX | <b>MINIMUM/MAXIMUM DISPLAY MODE:</b><br>0: LED display set in minimum display mode.<br>1: LED display set in maximum display mode.                                                                                                                                                                                                                                                    |

## 8.0 RIC Registers (Continued)

### Real Time Interrupt Register (Address 0FH)

The Real Time Interrupt register (RTI) contains information which may change on a packet by packet basis. Any remaining interrupts which have not been serviced before the following packet is transmitted are cleared. Since multiple interrupt sources may be displayed by the RTI a priority scheme is implemented. A read cycle to the RTI gives the interrupt source and an address vector indicating the RIC port which generated the interrupt. The order of priority for the display of interrupt information is as follows:

1. The receive source of network activity (Port N),
2. The first RIC port showing collision
3. A port partitioned or reconnected.

During the repetition of a single packet it is possible that multiple ports may be partitioned or alternatively reconnected. The ports have equal priority in displaying partition/reconnection information. This data is derived from the ports by the RTI register as it polls consecutively around the ports.

Reading the RTI clears the particular interrupt. If no interrupt sources are active the RTI returns a no valid interrupt status.

|        |        |        |        |       |       |       |       |
|--------|--------|--------|--------|-------|-------|-------|-------|
| D7     | D6     | D5     | D4     | D3    | D2    | D1    | D0    |
| IVCTR3 | IVCTR2 | IVCTR1 | IVCTR0 | ISRC3 | ISRC2 | ISRC1 | ISRC0 |

| Bit    | R/W | Symbol Access | Description                                                                                            |
|--------|-----|---------------|--------------------------------------------------------------------------------------------------------|
| D(3:0) | R   | ISCR(3:0)     | <b>INTERRUPT SOURCE:</b> These four bits indicate the reason why the interrupt was generated.          |
| D(7:4) | R   | IVCTR(3:0)    | <b>INTERRUPT VECTOR:</b> This field defines the port address responsible for generating the interrupt. |

The following table shows the mapping of interrupt sources onto the D3 to D0 pins. Essentially each of the three interrupt sources has a dedicated bit in this field. If a read to the RTI produces a low logic level on one of these bits then the interrupt source may be directly decoded. Associated with the source of the interrupt is the port where the event is occurring. If no unmasked events (receive, collision, etc.), have occurred when the RTI is read then an all ones pattern is driven by the RIC onto the data pins.

| D7  | D6  | D5  | D4  | D3 | D2 | D1 | D0 | Comments                                                   |
|-----|-----|-----|-----|----|----|----|----|------------------------------------------------------------|
| PA3 | PA2 | PA1 | PA0 | 1  | 1  | 0  | 1  | First Collision<br>PA(3:0) = Collision Port Address        |
| PA3 | PA2 | PA1 | PA0 | 1  | 0  | 1  | 1  | Receive<br>PA(3:0) = Receive Port Address                  |
| PA3 | PA2 | PA1 | PA0 | 0  | 1  | 1  | 1  | Partition Reconnection<br>PA(3:0) = Partition Port Address |
| 1   | 1   | 1   | 1   | 1  | 1  | 1  | 1  | No Valid Interrupt                                         |

## 8.0 RIC Registers (Continued)

### Page Select Register ((All Pages) Address 10H)

The Page Select register performs two functions:

1. It enables switches to be made between register pages,
2. It provides status information regarding the Event Logging Interrupts.

|    |    |    |    |      |       |       |       |
|----|----|----|----|------|-------|-------|-------|
| D7 | D6 | D5 | D4 | D3   | D2    | D1    | D0    |
| FC | HC | LC | FF | resv | PSEL2 | PSEL1 | PSEL0 |

| Bit    | R/W | Symbol    | Description                                                                                                                                                         |
|--------|-----|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D(2:0) | R/W | PSEL(2:0) | <b>PAGE SELECT BITS:</b> When read these bits indicate the currently selected Upper Register Array Page. Write cycles to these locations facilitates page swapping. |
| D3     | R   | resv      | <b>RESERVED FOR FUTURE USE</b>                                                                                                                                      |
| D4     | R   | FF        | <b>FLAG FOUND:</b> This indicates one of the unmasked event recording latches has been set.                                                                         |
| D5     | R   | LC        | <b>LOW COUNT:</b> This indicates one of the port event counters has a value less than 00FF Hex.                                                                     |
| D6     | R   | HC        | <b>HIGH COUNT:</b> This indicates one of the port event counters has a value greater than C000 Hex.                                                                 |
| D7     | R   | FC        | <b>FULL COUNTER:</b> This indicates one of the port event counters has a value equal to FFFF Hex.                                                                   |

### Device Type Register (Page 0H Address 11H)

This register may be used to distinguish different revisions of RIC. If this register is read it will return a different value each for DP83950 revisions. (Contact National Semiconductor for revision information.) Write operations to this register have no effect upon the contents.

|    |    |    |    |    |    |    |    |
|----|----|----|----|----|----|----|----|
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| 0  | 0  | 0  | 0  | 0  | 0  | X  | X  |

## 8.0 RIC Registers (Continued)

### Lower Event Count Mask Register (Page 0H Address 12H)

| D7     | D6    | D5   | D4  | D3    | D2    | D1     | D0   |
|--------|-------|------|-----|-------|-------|--------|------|
| BDLNKC | PARTC | RECC | SEC | NSFDC | PLERC | ELBERC | JABC |

| Bit | R/W | Symbol | Description                                                                                                            |
|-----|-----|--------|------------------------------------------------------------------------------------------------------------------------|
| D0  | R/W | JABC   | <b>JABBER COUNT ENABLE:</b> Enables recording of Jabber Protect events.                                                |
| D1  | R/W | ELBERC | <b>ELASTICITY BUFFER ERROR COUNT ENABLE:</b> Enables recording of Elasticity Buffer Error events.                      |
| D2  | R/W | PLERC  | <b>PHASE LOCK ERROR COUNT ENABLE:</b> Enables recording of Carrier Error events.                                       |
| D3  | R/W | NSFDC  | <b>NON SFD COUNT ENABLE:</b> Enables recording of Non SFD packet events.                                               |
| D4  | R/W | SEC    | <b>SHORT EVENT COUNT ENABLE:</b> Enables recording of Short events.                                                    |
| D5  | R/W | RECC   | <b>RECEIVE COUNT ENABLE:</b> Enables recording of Packet Receive (port N status) events that do not suffer collisions. |
| D6  | R/W | PARTC  | <b>PARTITION COUNT ENABLE:</b> Enables recording of Partition events.                                                  |
| D7  | R/W | BDLNKC | <b>BAD LINK COUNT ENABLE:</b> Enables recording of Bad Link events.                                                    |

### Upper Event Count Mask Register (Page 0H Address 13H)

| D7   | D6   | D5   | D4     | D3     | D2   | D1  | D0  |
|------|------|------|--------|--------|------|-----|-----|
| resv | resv | OWCC | RXCOLC | TXCOLC | resv | FWF | ROR |

| Bit    | R/W | Symbol | Description                                                                                                                                                                      |
|--------|-----|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0     | R/W | ROR    | <b>RESET ON READ:</b> This bit selects the action a read operation has upon a port's event counter:<br>0: No effect upon register contents.<br>1: The counter register is reset. |
| D1     | R/W | FWF    | <b>FREEZE WHEN FULL:</b> This bit controls the freezing of the Event Count registers when the counter is full (FFFF Hex)                                                         |
| D2     | R   | resv   | <b>RESERVED FOR FUTURE USE:</b> This bit should be written with a low logic level.                                                                                               |
| D3     | R/W | TXCOLC | <b>TRANSMIT COLLISION COUNT ENABLE:</b> Enables recording of transmit collision events only.                                                                                     |
| D4     | R/W | RXCOLC | <b>RECEIVE COLLISION COUNT ENABLE:</b> Enables recording of receive collision events only.                                                                                       |
| D5     | R/W | OWCC   | <b>OUT OF WINDOW COLLISION COUNT ENABLE:</b> Enables recording of out of window collision events only.                                                                           |
| D(7:6) | R   | resv   | <b>RESERVED FOR FUTURE USE:</b> These bits should be written with a low logic level.                                                                                             |

**Note 1:** To count all collisions then both the TXCOLC and RXCOLC bits must be set. The OWCC bit should not be set otherwise the port counter will be incremented twice when an out of collision window collision occurs. The OWCC bit alone should be set if only out of window collision are to be counted.

**Note 2:** Writing a 1 enables the event to be counted.



## 8.0 RIC Registers (Continued)

### Event Record Mask Register (Page 0H Address 14H)

| D7     | D6    | D5   | D4  | D3    | D2    | D1     | D0   |
|--------|-------|------|-----|-------|-------|--------|------|
| BDLNKE | PARTE | OWCE | SEE | NSFDE | PLERE | ELBERE | JABE |

| Bit | R/W | Symbol | Description                                                                                            |
|-----|-----|--------|--------------------------------------------------------------------------------------------------------|
| D0  | R/W | JABE   | <b>JABBER ENABLE:</b> Enables recording of Jabber Protect events.                                      |
| D1  | R/W | ELBERE | <b>ELASTICITY BUFFER ERROR ENABLE:</b> Enables recording of Elasticity Buffer Error events.            |
| D2  | R/W | PLERE  | <b>PHASE LOCK ERROR ENABLE:</b> Enables recording of Carrier Error events.                             |
| D3  | R/W | NSFDE  | <b>NON SFD ENABLE:</b> Enables recording of Non SFD packet events.                                     |
| D4  | R/W | SEE    | <b>SHORT EVENT ENABLE:</b> Enables recording of Short Events.                                          |
| D5  | R/W | OWCE   | <b>OUT OF WINDOW COLLISION COUNT ENABLE:</b> Enables recording of Out of Window Collision events only. |
| D6  | R/W | PARTE  | <b>PARTITION ENABLE:</b> Enables recording of Partition events.                                        |
| D7  | R/W | BDLNKE | <b>BAD LINK ENABLE:</b> Enables recording of Bad Link Events.                                          |

**Note:** Writing a 1 enables the event to be recorded.

## 8.0 RIC Registers (Continued)

### Interrupt and Management Configuration Register (Page 0H Address 16H)

This register powers up with all bits set to one and must be initialized by a processor write cycle before any events will generate interrupts.

|                         |                         |                         |                         |                          |                          |                           |        |
|-------------------------|-------------------------|-------------------------|-------------------------|--------------------------|--------------------------|---------------------------|--------|
| D7                      | D6                      | D5                      | D4                      | D3                       | D2                       | D1                        | D0     |
| $\overline{\text{IFC}}$ | $\overline{\text{IHC}}$ | $\overline{\text{ILC}}$ | $\overline{\text{IFF}}$ | $\overline{\text{IREC}}$ | $\overline{\text{ICOL}}$ | $\overline{\text{IPART}}$ | MIFCON |

| Bit | R/W | Symbol                    | Description                                                                                                                                                                                                                                                                                                                                                              |
|-----|-----|---------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0  | R/W | MIFCON                    | <b>MANAGEMENT INTERFACE CONFIGURATION:</b><br>0: All Packets repeated are transmitted over the Management bus.<br>1: Packets repeated by the RIC which do not have a Start of Frame Delimiters are not transmitted over the Management bus.                                                                                                                              |
| D1  | R/W | $\overline{\text{IPART}}$ | <b>INTERRUPT ON PARTITION:</b><br>0: Interrupts will be generated <sup>(1)</sup> if a port becomes Partitioned.<br>1: No interrupts are generated by this condition.                                                                                                                                                                                                     |
| D2  | R/W | $\overline{\text{ICOL}}$  | <b>INTERRUPT ON COLLISION:</b><br>0: Interrupts will be generated <sup>(1)</sup> if this RIC has a port which experiences a collision, Single RIC applications, or contains a port which experiences a receive collision or is the first port to suffer a transmit collision in a packet in Multi-RIC applications.<br>1: No interrupts are generated by this condition. |
| D3  | R/W | $\overline{\text{IREC}}$  | <b>INTERRUPT ON RECEIVE:</b><br>0: Interrupts will be generated <sup>(1)</sup> if this RIC contains the receive port for packet or collision activity.<br>1: No interrupts are generated by this condition.                                                                                                                                                              |
| D4  | R/W | $\overline{\text{IFF}}$   | <b>INTERRUPT ON FLAG FOUND:</b><br>0: Interrupts will be generated <sup>(2)</sup> if one or more than one of the flags in the flag array is true.<br>1: No interrupts are generated by this condition.                                                                                                                                                                   |
| D5  | R/W | $\overline{\text{ILC}}$   | <b>INTERRUPT ON LOW COUNT:</b><br>0: Interrupt generated <sup>(2)</sup> when one or more of the Event Counters holds a value less than 256 counts.<br>1: No effect                                                                                                                                                                                                       |
| D6  | R/W | $\overline{\text{IHC}}$   | <b>INTERRUPT ON HIGH COUNT:</b><br>0: Interrupt generated <sup>(2)</sup> when one or more of the Event Counters holds a value in excess of 49152 counts.<br>1: No effect                                                                                                                                                                                                 |
| D7  | R/W | $\overline{\text{IFC}}$   | <b>INTERRUPT ON FULL COUNTER:</b><br>0: Interrupt generated <sup>(2)</sup> when one or more of the Event Counters is full.<br>1: No effect                                                                                                                                                                                                                               |

**Note 1:** (RTI pin goes active)

**Note 2:** (ELI pin goes active)

## 8.0 RIC Registers (Continued)

### RIC Address Register (Page 0H Address 17H)

This register may be used to differentiate between RICs in a multi-RIC repeater system. The contents of this register form part of the information available through the management bus.

|    |    |    |    |    |    |     |     |
|----|----|----|----|----|----|-----|-----|
| D7 | D6 | D5 | D4 | D3 | D2 | D1  | D0  |
| A5 | A4 | A3 | A2 | A1 | A0 | res | res |

### Packet Compress Decode Register (Page 0H Address 18H)

This register is used to determine the number of bytes in the data field of a packet which are transferred over the management bus when the packet compress option is employed. The register bits perform the function of a direct binary decode. Thus up to 255 bytes of data may be transferred over the management bus if packet compression is selected.

|      |      |      |      |      |      |      |      |
|------|------|------|------|------|------|------|------|
| D7   | D6   | D5   | D4   | D3   | D2   | D1   | D0   |
| PCD7 | PCD6 | PCD5 | PCD4 | PCD3 | PCD2 | PCD1 | PCD0 |

### Inter Frame Gap Threshold Select Register (Page 0H Address 1FH)

This register is used to configure the hub management interface to provide a certain minimum inter frame gap between packets transmitted over the management bus. The value written to this register, plus one, is the magnitude in bit times of the minimum IFG allowed on the management bus.

|       |       |       |       |       |       |       |       |
|-------|-------|-------|-------|-------|-------|-------|-------|
| D7    | D6    | D5    | D4    | D3    | D2    | D1    | D0    |
| IFGT7 | IFGT6 | IFGT5 | IFGT4 | IFGT3 | IFGT2 | IFGT1 | IFGT0 |

### Port Event Record Registers (Page 1H Address 11H to 1DH)

These registers hold the recorded events for the specified RIC port. The flags are cleared when the register is read.

|       |      |     |    |      |      |       |     |
|-------|------|-----|----|------|------|-------|-----|
| D7    | D6   | D5  | D4 | D3   | D2   | D1    | D0  |
| BDLNK | PART | OWC | SE | NSFD | PLER | ELBER | JAB |

| Bit | R/W | Symbol | Description                                                                    |
|-----|-----|--------|--------------------------------------------------------------------------------|
| D0  | R   | JAB    | <b>JABBER:</b> A Jabber Protect event has occurred.                            |
| D1  | R   | ELBER  | <b>ELASTICITY BUFFER ERROR:</b> A Elasticity Buffer Error has occurred.        |
| D2  | R   | PLER   | <b>PHASE LOCK ERROR:</b> A Phase Lock Error event has occurred.                |
| D3  | R   | NSFD   | <b>NON SFD:</b> A Non SFD packet event has occurred.                           |
| D4  | R   | SE     | <b>SHORT EVENT:</b> A Short event has occurred.                                |
| D5  | R   | OWC    | <b>OUT OF WINDOW COLLISION:</b> An out of window collision event has occurred. |
| D6  | R   | PART   | <b>PARTITION:</b> A partition event has occurred.                              |
| D7  | R   | BDLNK  | <b>BAD LINK:</b> A link failure event has occurred.                            |

### Port Event Count Register (Pages 2H and 3H)

The Event Count (EC) register shows the instantaneous value of the specified port's 16-bit counter. The counter increments when an enabled event occurs. The counter may be cleared when it is read and prevented from rolling over when the maximum count is reached by setting the appropriate control bits in the Upper Event Count mask register. Since the RIC's processor port is octal and the counters are 16 bits long a temporary holding register is employed for register reads. When one of the counters is read, either high or low byte first, all 16 bits of the counter are transferred to a holding register. Provided the next read cycle to the counter array accesses the same counter's, other byte, then the read cycle accesses the holding register. This avoids the problem of events occurring in between the two processor reads and indicating a false count value. In order to enter a new value to the holding register a different counter must be accessed or the same counter byte must be re-read.

Lower Byte

|     |     |     |     |     |     |     |     |
|-----|-----|-----|-----|-----|-----|-----|-----|
| D7  | D6  | D5  | D4  | D3  | D2  | D1  | D0  |
| EC7 | EC6 | EC5 | EC4 | EC3 | EC2 | EC1 | EC0 |

Upper Byte

|      |      |      |      |      |      |     |     |
|------|------|------|------|------|------|-----|-----|
| D7   | D6   | D5   | D4   | D3   | D2   | D1  | D0  |
| EC15 | EC14 | EC13 | EC12 | EC11 | EC10 | EC9 | EC8 |

## 9.0 AC and DC Specifications

### Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

|                                 |                          |
|---------------------------------|--------------------------|
| Supply Voltage ( $V_{CC}$ )     | 0.5V to 7.0V             |
| DC Input Voltage ( $V_{IN}$ )   | -0.5V to $V_{CC} + 0.5V$ |
| DC Output Voltage ( $V_{OUT}$ ) | -0.5V to $V_{CC} + 0.5V$ |

|                                                       |                 |
|-------------------------------------------------------|-----------------|
| Storage Temperature Range ( $T_{STG}$ )               | -65°C to +150°C |
| Power Dissipation ( $P_D$ )                           | 2W              |
| Lead Temperature ( $T_L$ )<br>(Soldering, 10 seconds) | 260°C           |
| ESD Rating<br>( $R_{zap} = 1.5k, C_{zap} = 120 pF$ )  | 1500V           |

**DC Specifications**  $T_A = 0^\circ\text{C}$  to  $+70^\circ\text{C}$ ,  $V_{CC} = 5V \pm 5\%$  unless otherwise specified

#### PROCESSOR, LED, TWISTED PAIR PORTS, INTER-RIC AND MANAGEMENT INTERFACES

| Symbol   | Description                              | Conditions                                   | Min  | Max | Units         |
|----------|------------------------------------------|----------------------------------------------|------|-----|---------------|
| $V_{OH}$ | Minimum High Level Output Voltage        | $I_{OH} = -8 \text{ mA}$                     | 3.5  |     | V             |
| $V_{OL}$ | Minimum Low Level Output Voltage         | $I_{OL} = 8 \text{ mA}$                      |      | 0.4 | V             |
| $V_{IH}$ | Minimum High Level Input Voltage         |                                              | 2.0  |     | V             |
| $V_{IL}$ | Maximum Low Level Input Voltage          |                                              |      | 0.8 | V             |
| $I_{IN}$ | Input Current                            | $V_{IN} = V_{CC}$ or GND                     | -1.0 | 1.0 | $\mu\text{A}$ |
| $I_{OZ}$ | Maximum TRI-STATE Output Leakage Current | $V_{OUT} = V_{CC}$ or GND                    | -10  | 10  | $\mu\text{A}$ |
| $I_{CC}$ | Average Supply Current                   | $V_{IN} = V_{CC}$ or GND<br>$V_{CC} = 5.25V$ |      | 380 | mA            |

#### AUI (PORT 1)

|          |                                                                          |                                                    |                |            |    |
|----------|--------------------------------------------------------------------------|----------------------------------------------------|----------------|------------|----|
| $V_{OD}$ | Differential Output Voltage ( $TX_{\pm}$ )                               | 78 $\Omega$ Termination and 270 $\Omega$ Pulldowns | $\pm 550$      | $\pm 1200$ | mV |
| $V_{OB}$ | Differential Output Voltage Imbalance ( $TX_{\pm}$ )                     | 78 $\Omega$ Termination and 270 $\Omega$ Pulldowns | Typical: 40 mV |            |    |
| $V_U$    | Undershoot Voltage ( $TX_{\pm}$ )                                        | 78 $\Omega$ Termination and 270 $\Omega$ Pulldowns | Typical: 80 mV |            |    |
| $V_{DS}$ | Differential Squelch Threshold ( $RX_{\pm}, CD_{\pm}$ )                  |                                                    | -175           | -300       | mV |
| $V_{CM}$ | Differential Input Common Mode Voltage ( $RX_{\pm}, CD_{\pm}$ ) (Note 1) |                                                    | 0              | 5.5        | V  |

**Note 1:** This parameter is guaranteed by design and is not tested.

## 9.0 AC and DC Specifications (Continued)

### DC Specifications $T_A = 0^\circ\text{C}$ to $+70^\circ\text{C}$ , $V_{CC} = 5\text{V} \pm 5\%$ unless otherwise specified (Continued)

| Symbol                         | Description                                                            | Conditions                                          | Min            | Max        | Units |
|--------------------------------|------------------------------------------------------------------------|-----------------------------------------------------|----------------|------------|-------|
| <b>PSEUDO AUI (PORTS 2–13)</b> |                                                                        |                                                     |                |            |       |
| $V_{POD}$                      | Differential Output Voltage (TX $\pm$ )                                | 270 $\Omega$ Termination and 1 k $\Omega$ Pulldowns | $\pm 450$      | $\pm 1200$ | mV    |
| $V_{POB}$                      | Differential Output Voltage Imbalance (TX $\pm$ )                      | 270 $\Omega$ Termination and 1 k $\Omega$ Pulldowns | Typical: 40 mV |            |       |
| $V_{PU}$                       | Undershoot Voltage (TX $\pm$ )                                         | 270 $\Omega$ Termination and 1 k $\Omega$ Pulldowns | Typical: 80 mV |            |       |
| $V_{PDS}$                      | Differential Squelch Threshold (RX $\pm$ , CD $\pm$ )                  |                                                     | -175           | -300       | mV    |
| $V_{PCM}$                      | Differential Input Common Mode Voltage (Rx $\pm$ , CD $\pm$ ) (Note 1) |                                                     | 0              | 5.5        | V     |

### TWISTED PAIR (PORTS 2–13)

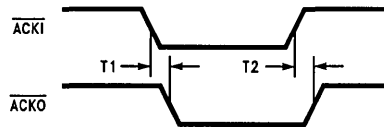
| Symbol    | Minimum Receive Squelch Threshold | Normal Mode<br>Reduced Mode | $\pm 300$<br>(Note 2) | $\pm 585$<br>$\pm 340$ | mV<br>mV |
|-----------|-----------------------------------|-----------------------------|-----------------------|------------------------|----------|
| $V_{RON}$ |                                   |                             |                       |                        |          |

**Note 1:** This parameter is guaranteed by design and is not tested.

**Note 2:** The operation in Reduced Mode is not guaranteed below 300 mV.

## AC Specifications

### PORT ARBITRATION TIMING



TL/F/11096-22

| Number | Symbol     | Parameter                                  | Min | Max | Units |
|--------|------------|--------------------------------------------|-----|-----|-------|
| T1     | ackilackol | ACKI Low to $\overline{\text{ACKO}}$ Low   |     | 24  | ns    |
| T2     | ackihackoh | ACKI High to $\overline{\text{ACKO}}$ High |     | 21  | ns    |

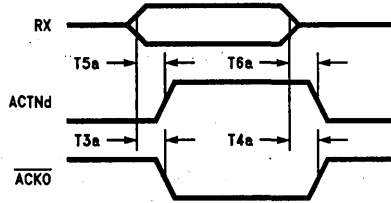
**Note:** Timing valid with no receive or collision activities.

**Note:** In these diagrams the Inter-RIC and Management Busses are shown using active high signals, active low signals may also be used. See Section 5.5 Mode Load Operation.

## 9.0 AC and DC Specifications (Continued)

### RECEIVING TIMINGS—AUI PORTS

Receive activity propagation start up and end delays for ports in non 10BASE-T mode



TL/F/11096-23

| Number | Symbol  | Parameter                                    | Min | Max | Units |
|--------|---------|----------------------------------------------|-----|-----|-------|
| T3a    | rxackol | RX Active to $\overline{\text{ACKO}}$ Low    |     | 66  | ns    |
| T4a    | rxackoh | RX Inactive to $\overline{\text{ACKO}}$ High |     | 325 | ns    |
| T5a    | rxactna | RX Active to ACTNd Active                    |     | 105 | ns    |
| T6a    | rxactni | RX Inactive to ACTNd Inactive                |     | 325 | ns    |

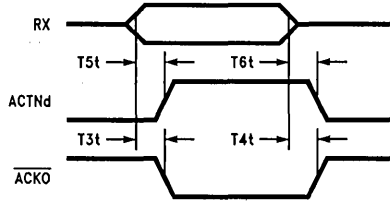
Note:  $\overline{\text{ACKO}}$  assumed high.

Note: In these diagrams the Inter-RIC and Management Busses are shown using active high signals, active low signals may also be used. See Section 5.5 Mode Load Operation.

## 9.0 AC and DC Specifications (Continued)

### RECEIVE TIMING—10BASE-T PORTS

Receive activity propagation start up and end delays for ports in 10BASE-T mode



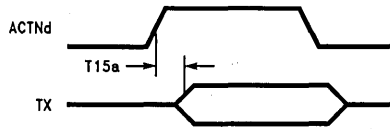
TL/F/11096-24

| Number | Symbol    | Parameter                                    | Min | Max | Units |
|--------|-----------|----------------------------------------------|-----|-----|-------|
| T3t    | rxaackol  | RX Active to $\overline{\text{ACKO}}$ Low    |     | 240 | ns    |
| T4t    | rxiaackoh | RX Inactive to $\overline{\text{ACKO}}$ High |     | 255 | ns    |
| T5t    | rxaactna  | RX Active to ACTNd Active                    |     | 270 | ns    |
| T6t    | rxiaactni | RX Inactive to ACTNd Inactive                |     | 265 | ns    |

Note:  $\overline{\text{ACKI}}$  assumed high.

### TRANSMIT TIMING—AUI PORTS

Transmit activity propagation start up and end delays for ports in non 10BASE-T mode



TL/F/11096-25

| Number | Symbol   | Parameter                 | Min | Max | Units |
|--------|----------|---------------------------|-----|-----|-------|
| T15a   | actnatxa | ACTNd Active to TX Active |     | 585 | ns    |

Note:  $\overline{\text{ACKI}}$  assumed high.

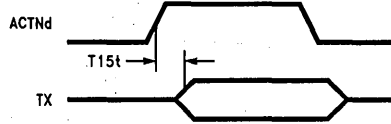
Note:  $\text{ACTNd}$  and  $\text{ACTNs}$  are tied together.

Note: In these diagrams the Inter-RIC and Management Busses are shown using active high signals, active low signals may also be used. See Section 5.5 Mode Load Operation.

## 9.0 AC and DC Specifications (Continued)

### TRANSMIT TIMING—10BASE-T PORTS

Receive activity propagation start up and end delays for ports in 10BASE-T mode



TL/F/11096-26

| Number | Symbol   | Parameter                 | Min | Max | Units |
|--------|----------|---------------------------|-----|-----|-------|
| T15t   | actnatxa | ACTNd Active to TX Active |     | 790 | ns    |

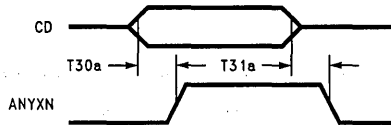
Note:  $\overline{ACKI}$  assumed high.

Note: ACTN<sub>d</sub> and ACTN<sub>s</sub> are tied together.

### COLLISION TIMING—AUI PORTS

Collision activity propagation start up and end delays for ports in non 10BASE-T mode

#### TRANSMIT COLLISION TIMING



TL/F/11096-27

| Number | Symbol    | Parameter                                  | Min | Max | Units |
|--------|-----------|--------------------------------------------|-----|-----|-------|
| T30a   | cdaanyxna | CD Active to ANYXN Active                  |     | 65  | ns    |
| T31a   | cdianyxn  | CD Inactive to ANYXN Inactive (Notes 1, 2) |     | 400 | ns    |

Note 1: TX collision extension has already been performed and no other port is driving ANYXN.

Note 2: Includes TW2.

Note: In these diagrams the Inter-RIC and Management Busses are shown using active high signals, active low signals may also be used. See Section 5.5 Mode Load Operation.

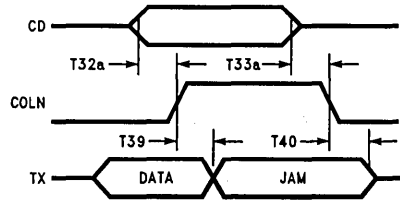


## 9.0 AC and DC Specifications (Continued)

### COLLISION TIMING—AUI PORTS

Collision activity propagation start up and end delays for ports in non 10BASE-T mode.

#### RECEIVE COLLISION TIMING



TL/F/11096-28

| Number | Symbol   | Parameter                            | Min | Max | Units |
|--------|----------|--------------------------------------|-----|-----|-------|
| T32a   | cdacolna | CD Active to COLN Active (Note 1)    |     | 55  | ns    |
| T33a   | cdicolni | CD Inactive to COLN Inactive         |     | 215 | ns    |
| T39    | colnajs  | COLN Active to Start of Jam          |     | 400 | ns    |
| T40    | colnije  | COLN Inactive to End of Jam (Note 2) |     | 800 | ns    |

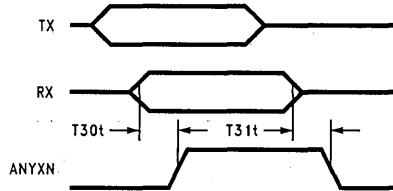
**Note 1:** PKEN assumed high.

**Note 2:** Assuming reception ended before COLN goes inactive. TW2 is included in this parameter. Assuming  $ACTN_d$  to  $ACTN_s$  delay is 0.

**Note:** In these diagrams the Inter-RIC and Management Busses are shown using active high signals, active low signals may also be used. See Section 5.5 Mode Load Operation.

### COLLISION TIMING—10BASE-T PORTS

Collision activity propagation start up and end delays for ports in 10BASE-T mode



TL/F/11096-29

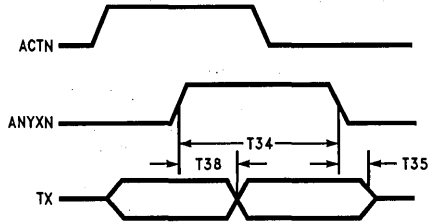
| Number | Symbol   | Parameter                                     | Min | Max | Units |
|--------|----------|-----------------------------------------------|-----|-----|-------|
| T30t   | colaanya | Collision Active to ANYXN Active              |     | 800 | ns    |
| T31t   | coliani  | Collision Inactive to ANYXN Inactive (Note 1) |     | 400 | ns    |

**Note 1:** TX collision extension has already been performed and no other port is asserting ANYXN.

**Note:** In these diagrams the Inter-RIC and Management Busses are shown using active high signals, active low signals may also be used. See Section 5.5 Mode Load Operation.

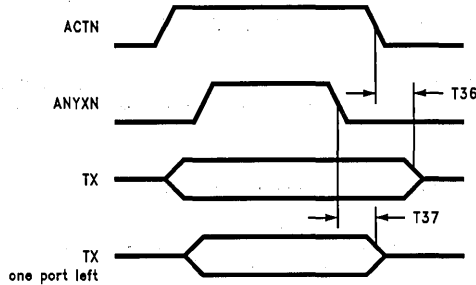
## 9.0 AC and DC Specifications (Continued)

### COLLISION TIMING—ALL PORTS



TL/F/11096-38

| Number | Symbol   | Parameter                            | Min | Max | Units |
|--------|----------|--------------------------------------|-----|-----|-------|
| T34    | anyamin  | ANYXN Active Time                    | 96  |     | Bits  |
| T35    | anyitxai | ANYXN Inactive to TX to all Inactive | 120 | 170 | ns    |
| T38    | anyasj   | ANYXN Active to Start of Jam         |     | 400 | ns    |



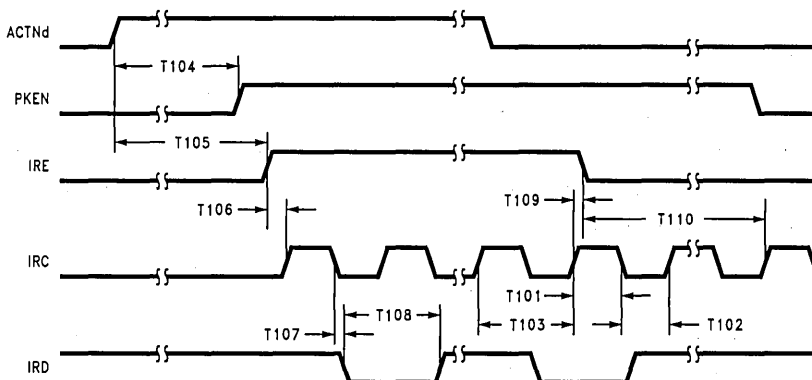
TL/F/11096-39

| Number | Symbol   | Parameter                                     | Min | Max | Units |
|--------|----------|-----------------------------------------------|-----|-----|-------|
| T36    | actnitxi | ACTN Inactive to TX Inactive                  |     | 405 | ns    |
| T37    | anyitxoi | ANYXN Inactive to TX "One Port Left" Inactive | 120 | 170 | ns    |

**Note:** In these diagrams the Inter-RIC and Management Busses are shown using active high signals, active low signals may also be used. See Section 5.5 Mode Load Operation.

## 9.0 AC and DC Specifications (Continued)

### INTER RIC BUS OUTPUT TIMING



TL/F/11096-35

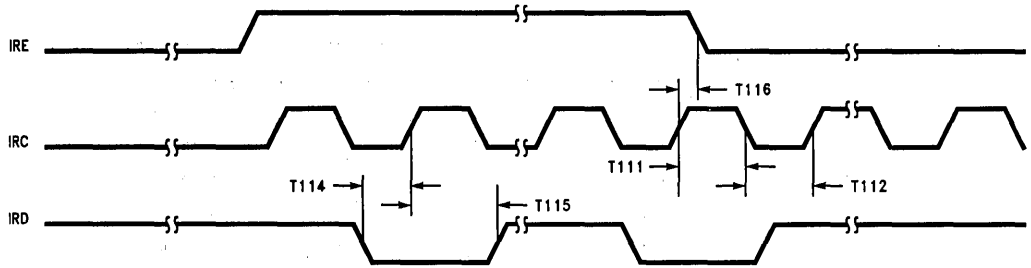
| Number | Symbol      | Parameter                         | Min | Max | Units   |
|--------|-------------|-----------------------------------|-----|-----|---------|
| T101   | ircoh       | IRC Output High Time              | 45  | 55  | ns      |
| T102   | ircol       | IRC Output Low Time               | 45  | 55  | ns      |
| T103   | ircoc       | IRC Output Cycle Time             | 90  | 110 | ns      |
| T104   | actndapkena | ACTNd Active to PKEN Active       | 555 |     | ns      |
| T105   | actndairea  | ACTNd Active to IRE Active        | 560 |     | ns      |
| T106   | ireoairca   | IRE Output Active to IRC Active   |     | 1.8 | $\mu$ s |
| T107   | irdov       | IRD Output Valid from IRC         |     | 10  | ns      |
| T108   | irdos       | IRD Output Stable Valid Time      | 90  |     | ns      |
| T109   | ircohirei   | IRC Output High to IRE Inactive   | 30  | 70  | ns      |
| T110   | ircclks     | Number of IRCs after IRE Inactive | 5   |     | clks    |

**Note:** In these diagrams the Inter-RIC and Management Busses are shown using active high signals, active low signals may also be used. See Section 5.5 Mode Load Operation.

**Note:** In a Multi-RIC system, the PKEN signal is valid only for the first receiving RIC.

## 9.0 AC and DC Specifications (Continued)

### INTER-RIC BUS INPUT TIMING



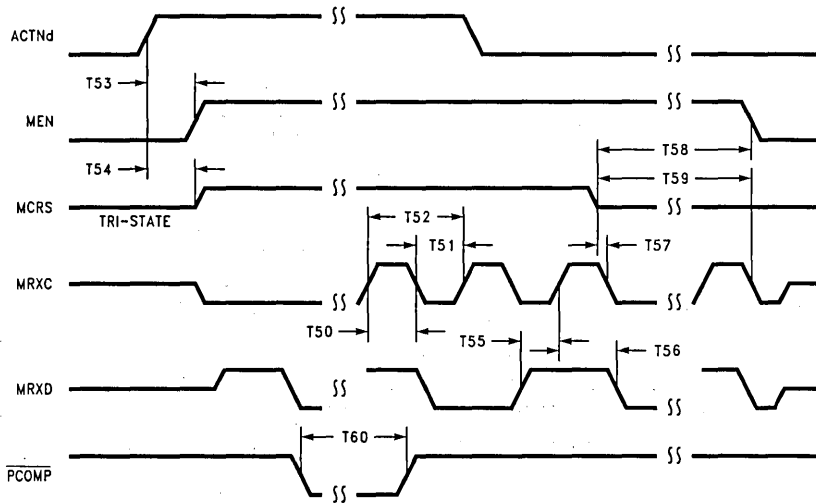
TL/F/11096-40

| Number | Symbol   | Parameter                      | Min | Max | Units |
|--------|----------|--------------------------------|-----|-----|-------|
| T111   | ircih    | IRC Input High Time            | 20  |     | ns    |
| T112   | ircil    | IRC Input Low Time             | 20  |     | ns    |
| T114   | irdisirc | IRD Input Setup to IRC         | 5   |     | ns    |
| T115   | irdihirc | IRD Input Hold from IRC        | 10  |     | ns    |
| T116   | irchiire | IRC Input High to IRE Inactive | 10  | 90  | ns    |

**Note:** In these diagrams the Inter-RIC and Management Busses are shown using active high signals, active low signals may also be used. See Section 5.5 Mode Load Operation.

## 9.0 AC and DC Specifications (Continued)

### MANAGEMENT BUS TIMING



TL/F/11096-30

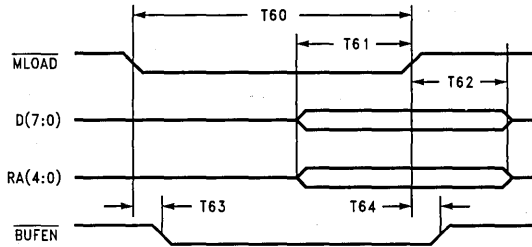
| Number | Symbol      | Parameter                               | Min | Max | Units |
|--------|-------------|-----------------------------------------|-----|-----|-------|
| T50    | mrxch       | MRXC High Time                          | 45  | 55  | ns    |
| T51    | mrxcl       | MRXC Low Time                           | 45  | 55  | ns    |
| T52    | mrxcd       | MRXC Cycle Time                         | 90  | 110 | ns    |
| T53    | actndamena  | ACTNd Active to MEN Active              |     | 715 | ns    |
| T54    | actndamcrsa | ACTNd Active to MCRS Active             |     | 720 | ns    |
| T55    | mrxd        | MRXD Setup                              | 40  |     | ns    |
| T56    | mrxdh       | MRXD Hold                               | 45  |     | ns    |
| T57    | mrxclmcrsi  | MRXC Low to MCRS Inactive               | -5  | 6   | ns    |
| T58    | mcrsimenl   | MCRS Inactive to MEN Low                |     | 510 | ns    |
| T59    | mrxcclks    | Min Number of MRXCs after MCRS Inactive | 5   | 5   | Clks  |
| T60    | pcompw      | PCOMP Pulse Width                       | 20  |     | ns    |

**Note:** The preamble on this bus consists of the following string: 01011.

**Note:** In these diagrams the Inter-RIC and Management Busses are shown using active high signals, active low signals may also be used. See Section 5.5 Mode Load Operation.

## 9.0 AC and DC Specifications (Continued)

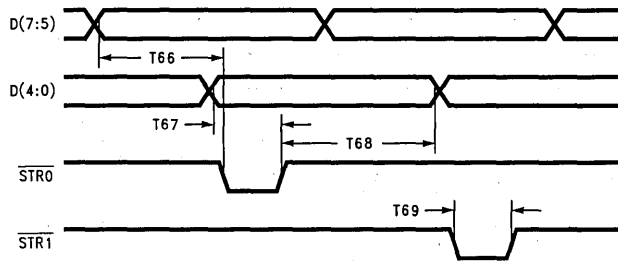
### MLOAD TIMING



TL/F/11096-31

| Number | Symbol  | Parameter                        | Min | Max | Units |
|--------|---------|----------------------------------|-----|-----|-------|
| T61    | mldats  | Data Setup                       | 10  |     | ns    |
| T62    | mldath  | Data Hold                        | 10  |     | ns    |
| T63    | mlabufa | MLOAD Active to BUFEN Active     |     | 35  | ns    |
| T64    | mlibufi | MLOAD Inactive to BUFEN Inactive |     | 35  | ns    |
| T65    | mlw     | MLOAD Width                      | 800 |     | ns    |

### STROBE TIMING



TL/F/11096-32

| Number | Symbol  | Parameter            | Min | Max | Units |
|--------|---------|----------------------|-----|-----|-------|
| T66    | stradsr | Strobe Address Setup | 80  | 115 | ns    |
| T67    | strdats | Strobe Data Setup    | 40  | 65  | ns    |
| T68    | strdath | Strobe Data Hold     | 135 | 160 | ns    |
| T69    | strw    | Strobe Width         | 30  | 65  | ns    |

### CDEC TIMING

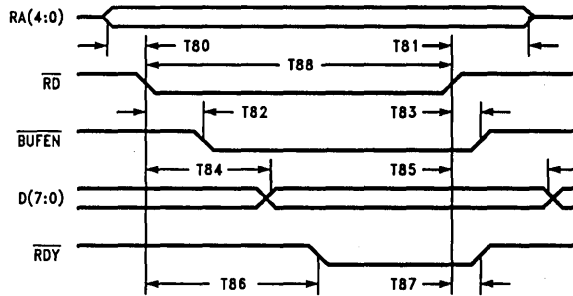


TL/F/11096-41

| Number | Symbol  | Parameter          | Min | Max | Units |
|--------|---------|--------------------|-----|-----|-------|
| T70    | cdecpw  | CDEC Pulse Width   | 20  | 100 | ns    |
| T71    | cdecdec | CDEC to CDEC Width | 200 |     | ns    |

## 9.0 AC and DC Specifications (Continued)

### REGISTER READ TIMING



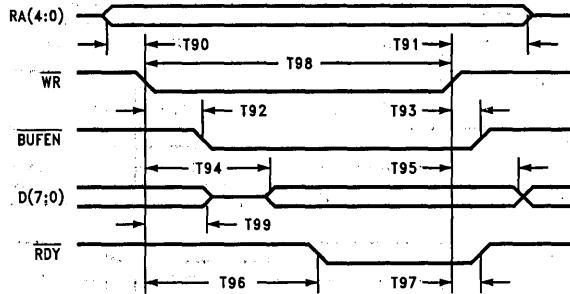
TL/F/11096-33

| Number | Symbol  | Parameter                       | Min | Max | Units |
|--------|---------|---------------------------------|-----|-----|-------|
| T80    | rdads   | Read Address Setup              | 0   |     | ns    |
| T81    | rdadrh  | Read Address Hold               | 0   |     | ns    |
| T82    | rdabufa | Read Active to BUFEN Active     | 95  | 345 | ns    |
| T83    | rdibufi | Read Inactive to BUFEN Inactive |     | 35  | ns    |
| T84    | rdadatv | Read Active to Data Invalid     | 245 |     | ns    |
| T85    | rdath   | Read Data Hold                  | 75  |     | ns    |
| T86    | rdardya | Read Active to RDY Active       | 340 | 585 | ns    |
| T87    | rdirdyi | Read Inactive to RDY Inactive   |     | 30  | ns    |
| T88    | rdw     | Read Width                      | 600 |     | ns    |

Note: Minimum high time between read/write cycles is 100 ns.

# 9.0 AC and DC Specifications (Continued)

## REGISTER WRITE TIMING



TL/F/11096-34

| Number | Symbol  | Parameter                                            | Min | Max | Units |
|--------|---------|------------------------------------------------------|-----|-----|-------|
| T90    | wradrs  | Write Address Setup                                  | 0   |     | ns    |
| T91    | wradrh  | Write Address Hold                                   | 0   |     | ns    |
| T92    | wrabufa | Write Active to $\overline{\text{BUFEN}}$ Active     | 95  | 355 | ns    |
| T93    | wribufi | Write Inactive to $\overline{\text{BUFEN}}$ Inactive |     | 35  | ns    |
| T94    | wradatv | Write Active to Data Valid                           |     | 275 | ns    |
| T95    | wrdath  | Write Data Hold                                      | 0   |     | ns    |
| T96    | wrardya | Write Active to $\overline{\text{RDY}}$ Active       | 340 | 585 | ns    |
| T97    | wrirnyi | Write Inactive to $\overline{\text{RDY}}$ Inactive   |     | 30  | ns    |
| T98    | wrw     | Write Width                                          | 600 |     | ns    |
| T99    | wradt   | Write Active to Data TRI-STATE                       |     | 350 | ns    |

Note: Assuming zero propagation delay on external buffer.

Note: Minimum high time between read/write cycles is 100 ns.

Note: The data will always TRI-STATE before  $\overline{\text{BUFEN}}$  goes active with a load of 100 pF on the data bus.

Note: When  $\overline{\text{RDY}}$  is used, the minimum 600 ns write width does not have to be maintained.

## 10.0 AC Timing Test Conditions

All specifications are valid only if the mandatory isolation is employed and all differential signals are taken to be at the AUI side of the pulse transformer.

Input Pulse Levels (TTL/CMOS) GND to 3.0V

Input Rise and Fall Times (TTL/CMOS) 5 ns

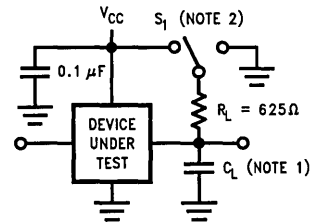
Input and Output Reference Levels (TTL/CMOS) 1.5V

Input Pulse Levels (Diff.) 2.0 V<sub>P-P</sub>

Input and Output Reference Levels (Diff.) 50% Point of the Differential

TRI-STATE Reference Levels Float ( $\Delta V$ )  $\pm 0.5V$

Output Load (See Figure Below)



TL/F/11096-36

Note 1: 100 pF, includes scope and jig capacitance.

Note 2: S1 = Open for timing tests for push pull outputs.

S1 = VCC for VOL test.

S1 = GND for VOH test.

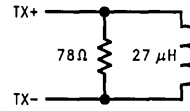
S1 = VCC for High Impedance to active low and active low to High Impedance measurements.

S1 = GND for High Impedance to active high and active high to High Impedance measurements.



**Capacitance**  $T_A = 25^\circ\text{C}, f = 1\text{ MHz}$ 

| Symbol    | Parameter          | Typ | Units |
|-----------|--------------------|-----|-------|
| $C_{IN}$  | Input Capacitance  | 7   | pF    |
| $C_{OUT}$ | Output Capacitance | 7   | pF    |



TL/F/11096-37

**Note:** In the above diagram, the TX+ and TX- signals are taken from the AUI side of the isolation (pulse transformer). The pulse transformer used for all testing is the Pulse Engineering PE64103.

## DP83952 Repeater Interface Controller with Security Features (RICTM II)

### General Description

The DP83952 RIC II Repeater Interface Controller is an "Enhanced" version of the DP83950 RIC. RIC II is fully backward pin and functional compatible with the RIC. The DP83952 RIC II has the same basic architecture as the RIC with additional feature enhancements. RIC II provides additional network security options, additional statistics for repeater activities, and a faster processor interface. When RIC II is used in a "non-secure" mode, it functions in the same manner as the DP83950 RIC. When RIC II is used in a "secure" mode, it restricts unauthorized nodes from intruding and/or eavesdropping into the network. The RIC II utilizes internal CAMs to store/compare addresses of valid nodes when network security is desired.

RIC II implements the IEEE 802.3 multiport repeater unit specifications. It is fully compliant with the 802.3 repeater specification for the repeater, segment partition, and jabber lockup protection state machines. (Continued)

### Features

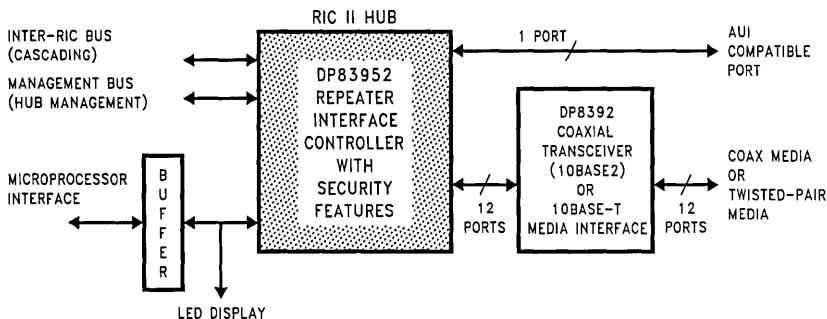
- Compliant with the IEEE 802.3 Repeater Specification
- 13 network connections (ports) per chip
- Selectable on-chip twisted-pair transceivers
- Cascadable for large hub applications
- Compatible with AUI compliant transceivers
- On-chip Elasticity Buffer, Manchester encoder and decoder

- Separate Partition state machines for each port
- Compatible with 802.3k Hub Management requirements
- Provides port status information for LED displays including: receive, collision, partition, link status, and jabber
- Power-up configuration options: Repeater and Partition Specifications, Transceiver Interface, Status Display, Processor Operations
- Simple processor interface for repeater management and port disable
- On-chip Event Counters and Event Flag Arrays
- Serial Management Bus Interface to combine packet and repeater status information
- CMOS process for low power dissipation
- Single 5V supply

### Security Features

- Power-up configuration options
- Prevents unauthorized eavesdropping and/or intrusion on a per port basis
- 58 on-chip CAMs (Content Addressable Memory) allow storage of acceptable addresses
- Learn mode automatically records addresses of attached nodes

### 1.0 System Diagram



TL/F/12499-1

## General Description (Continued)

The RIC II repeater design consists of two major functional blocks: Segment Specific Block, and Shared Functional Blocks. The Segment Specific Block implements the IEEE repeater requirements on a per network port basis, while the Shared Functional Blocks implement the core logic blocks for the IEEE repeater unit. The Shared Functional Blocks consist of repeater receive multiplexor, an on chip phase lock loop (PLL) decoder for Manchester data, an Elasticity Buffer for preamble regeneration, transmit encoder and demultiplexor for Manchester data.

The DP83952 RIC II can be connected up to 13 cable segments via its network interface ports. One port is fully AUI compatible and is able to connect to an external MAU using the maximum length of AUI cable. The other 12 ports have integrated 10BASE-T transceivers. These transceiver functions may be bypassed so that the RIC II may be used with external transceivers, such as the DP8392 coaxial transceivers.

A large repeater unit can be constructed by cascading RIC IIs together via the Inter-RIC™ bus. All the cascaded RIC IIs form a single repeater unit.

The RIC II is configurable for specific applications. It provides port status information for LED array displays, and a simple interface for system processors. The RIC II possesses multi-function counters and status flag arrays to facilitate network statistics gathering. A serial Hub Management Interface is available for the collection of data in Managed Hub applications.

## Table Of Contents

|                                                                            |  |
|----------------------------------------------------------------------------|--|
| <b>1.0 SYSTEM DIAGRAM</b>                                                  |  |
| <b>2.0 CONNECTION DIAGRAMS</b>                                             |  |
| <b>3.0 PIN DESCRIPTION</b>                                                 |  |
| <b>4.0 BLOCK DIAGRAM</b>                                                   |  |
| <b>5.0 FUNCTIONAL DESCRIPTION</b>                                          |  |
| 5.1 Summary of DP83952 RIC II Feature Enhancements from DP83950B RIC       |  |
| 5.2 Overview of RIC II Functions                                           |  |
| 5.3 Description of Repeater Operations                                     |  |
| 5.4 Examples of Packet Repetition Scenarios                                |  |
| 5.5 Description of Hardware Connection for Inter-RIC Bus                   |  |
| 5.6 Processor and Display Interface                                        |  |
| 5.7 Description of Hardware Connection for Processor and Display Interface |  |
| <b>6.0 HUB MANAGEMENT SUPPORT</b>                                          |  |
| 6.1 Event Counting Function                                                |  |
| 6.2 Event Record Function                                                  |  |
| 6.3 Management Interface Operation                                         |  |
| 6.4 Description of Hardware Connection for Management Interface            |  |
| <b>7.0 PORT BLOCK FUNCTIONS</b>                                            |  |
| 7.1 Transceiver Functions                                                  |  |
| 7.2 Segment Partition                                                      |  |
| 7.3 Port Status Register Functions                                         |  |
| 7.4 Local Ports and Inter-RIC Bus Expected Activity                        |  |
| 7.5 Local Ports and Inter-RIC Bus Data Field Contents                      |  |
| <b>8.0 RIC II REGISTERS</b>                                                |  |
| <b>9.0 AC AND DC SPECIFICATIONS</b>                                        |  |
| <b>10.0 TIMING AND LOAD DIAGRAMS</b>                                       |  |
| <b>11.0 AC TIMING TEST CONDITIONS</b>                                      |  |

## 2.0 Connection Diagrams

Pin Table (12 T.P. Ports + 1 AUI Bottom View)

| Pin Name        | Pin No. |
|-----------------|---------|
| TXO12P-         | 40      |
| TXO12+          | 39      |
| TXO12-          | 38      |
| TXO12P+         | 37      |
| RXI12-          | 36      |
| RXI12+          | 35      |
| V <sub>CC</sub> | 34      |
| GND             | 33      |
| RXI11-          | 32      |
| RXI11+          | 31      |
| TXO11P+         | 30      |
| TXO11-          | 29      |
| TXO11+          | 28      |
| TXO11P-         | 27      |
| V <sub>CC</sub> | 26      |
| GND             | 25      |
| TXO10P-         | 24      |
| TXO10+          | 23      |
| TXO10-          | 22      |
| TXO10P+         | 21      |
| RXI10-          | 20      |
| RXI10+          | 19      |
| V <sub>CC</sub> | 18      |
| GND             | 17      |
| RXI9-           | 16      |
| RXI9+           | 15      |
| TXO9P+          | 14      |
| TXO9-           | 13      |
| TXO9+           | 12      |
| TXO9P-          | 11      |
| V <sub>CC</sub> | 10      |
| GND             | 9       |
| TXO8P-          | 8       |
| TXO8+           | 7       |
| TXO8-           | 6       |
| TXO8P+          | 5       |
| RXI8-           | 4       |
| RXI8+           | 3       |
| V <sub>CC</sub> | 2       |
| GND             | 1       |

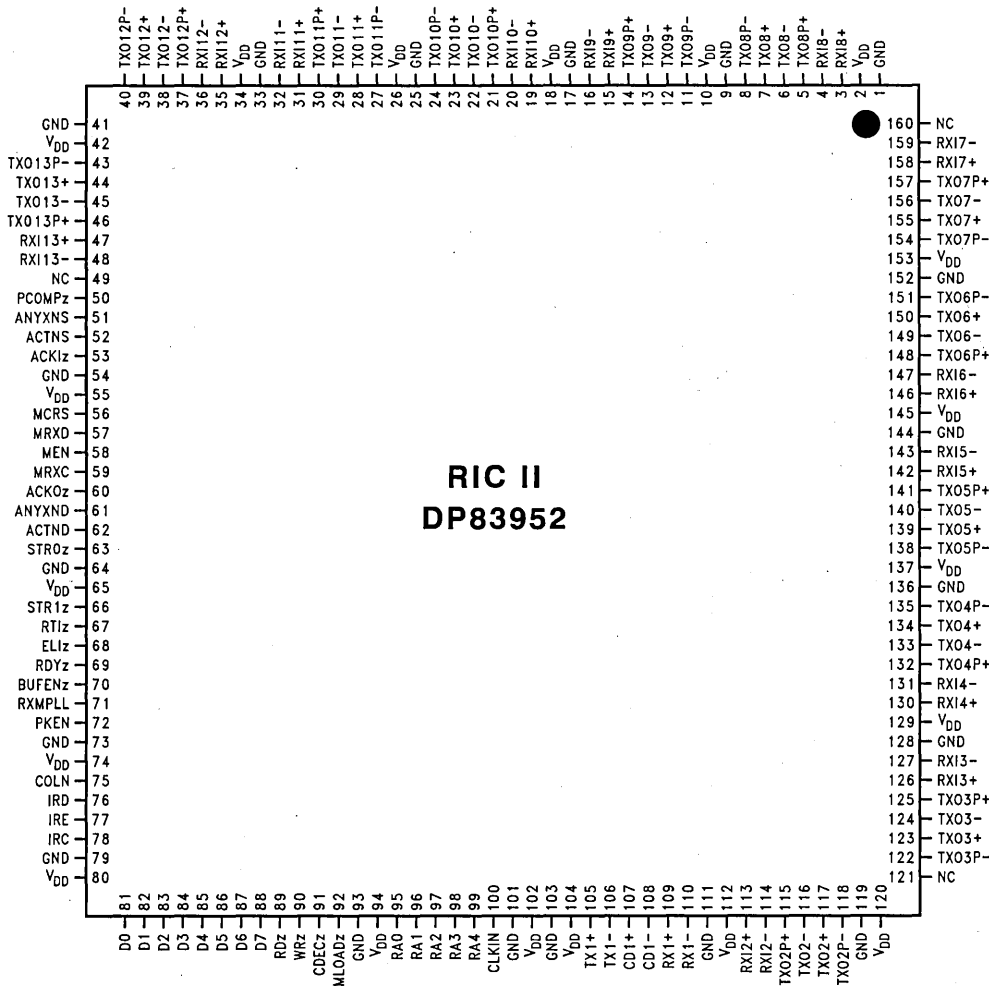
| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | 80      |
| GND             | 79      |
| IRC             | 78      |
| IRE             | 77      |
| IRD             | 76      |
| COLN            | 75      |
| V <sub>CC</sub> | 74      |
| GND             | 73      |
| PKEN            | 72      |
| RXMPLL          | 71      |
| BUFEN           | 70      |
| RDY             | 69      |
| ELI             | 68      |
| RTI             | 67      |
| STR1            | 66      |
| V <sub>CC</sub> | 65      |
| GND             | 64      |
| STR0            | 63      |
| ACTND           | 62      |
| ANYXND          | 61      |
| ACK0            | 60      |
| MRXC            | 59      |
| MEN             | 58      |
| MRXD            | 57      |
| MCRS            | 56      |
| V <sub>CC</sub> | 55      |
| GND             | 54      |
| ACK1            | 53      |
| ACTNS           | 52      |
| ANYXNS          | 51      |
| PCOMP           | 50      |
| NC              | 49      |
| RXI13-          | 48      |
| RXI13+          | 47      |
| TXO13P+         | 46      |
| TXO13-          | 45      |
| TXO13+          | 44      |
| TXO13P-         | 43      |
| V <sub>CC</sub> | 42      |
| GND             | 41      |

| Pin Name            | Pin No. |
|---------------------|---------|
| V <sub>CC</sub>     | 120     |
| GND                 | 119     |
| TXO2P-              | 118     |
| TXO2+               | 117     |
| TXO2-               | 116     |
| TXO2P+              | 115     |
| RXI2-               | 114     |
| RXI2+               | 113     |
| V <sub>CC</sub>     | 112     |
| GND                 | 111     |
| RX1-                | 110     |
| RX1+                | 109     |
| CD1-                | 108     |
| CD1+                | 107     |
| TX1-                | 106     |
| TX1+                | 105     |
| V <sub>CC</sub>     | 104     |
| GND                 | 103     |
| V <sub>CC</sub>     | 102     |
| GND                 | 101     |
| CLKIN               | 100     |
| PA4                 | 99      |
| PA3                 | 98      |
| PA2                 | 97      |
| PA1                 | 96      |
| PA0                 | 95      |
| V <sub>CC</sub> PLL | 94      |
| GNDPLL              | 93      |
| MLOAD               | 92      |
| ODEC                | 91      |
| WR                  | 90      |
| RD                  | 89      |
| D7                  | 88      |
| D6                  | 87      |
| D5                  | 86      |
| D4                  | 85      |
| D3                  | 84      |
| D2                  | 83      |
| D1                  | 82      |
| D0                  | 81      |

| Pin Name        | Pin No. |
|-----------------|---------|
| NC              | 160     |
| RXI7-           | 159     |
| RXI7+           | 158     |
| TXO7P+          | 157     |
| TXO7-           | 156     |
| TXO7+           | 155     |
| TXO7P-          | 154     |
| V <sub>CC</sub> | 153     |
| GND             | 152     |
| TXO6P-          | 151     |
| TXO6+           | 150     |
| TXO6-           | 149     |
| TXO6P+          | 148     |
| RXI6-           | 147     |
| RXI6+           | 146     |
| V <sub>CC</sub> | 145     |
| GND             | 144     |
| RXI5-           | 143     |
| RXI5+           | 142     |
| TXO5P+          | 141     |
| TXO5-           | 140     |
| TXO5+           | 139     |
| TXO5P-          | 138     |
| V <sub>CC</sub> | 137     |
| GND             | 136     |
| TXO4P-          | 135     |
| TXO4+           | 134     |
| TXO4-           | 133     |
| TXO4P+          | 132     |
| RXI4-           | 131     |
| RXI4+           | 130     |
| V <sub>CC</sub> | 129     |
| GND             | 128     |
| RXI3-           | 127     |
| RXI3+           | 126     |
| TXO3P+          | 125     |
| TXO3-           | 124     |
| TXO3+           | 123     |
| TXO3P-          | 122     |
| NC              | 121     |

Note: NC = No Connect

### 2.0 Connection Diagrams (Continued)



Ports 2-13 TP  
Port 1 AUI

TL/F/12499-2

## 2.0 Connection Diagrams (Continued)

Pin Table (1–5 AUI + 6–13 T.P. Ports)

| Pin Name        | Pin No. |
|-----------------|---------|
| TXO12P–         | 40      |
| TXO12+          | 39      |
| TXO12–          | 38      |
| TXO12P+         | 37      |
| RXI12–          | 36      |
| RXI12+          | 35      |
| V <sub>CC</sub> | 34      |
| GND             | 33      |
| RXI11–          | 32      |
| RXI11+          | 31      |
| TXO11P+         | 30      |
| TXO11–          | 29      |
| TXO11+          | 28      |
| TXO11P–         | 27      |
| V <sub>CC</sub> | 26      |
| GND             | 25      |
| TXO10P–         | 24      |
| TXO10+          | 23      |
| TXO10–          | 22      |
| TXO10P+         | 21      |
| RXI10–          | 20      |
| RXI10+          | 19      |
| V <sub>CC</sub> | 18      |
| GND             | 17      |
| RXI9–           | 16      |
| RXI9+           | 15      |
| TXO9P+          | 14      |
| TXO9–           | 13      |
| TXO9+           | 12      |
| TXO9P–          | 11      |
| V <sub>CC</sub> | 10      |
| GND             | 9       |
| TXO8P–          | 8       |
| TXO8+           | 7       |
| TXO8–           | 6       |
| TXO8P+          | 5       |
| RXI8–           | 4       |
| RXI8+           | 3       |
| V <sub>CC</sub> | 2       |
| GND             | 1       |

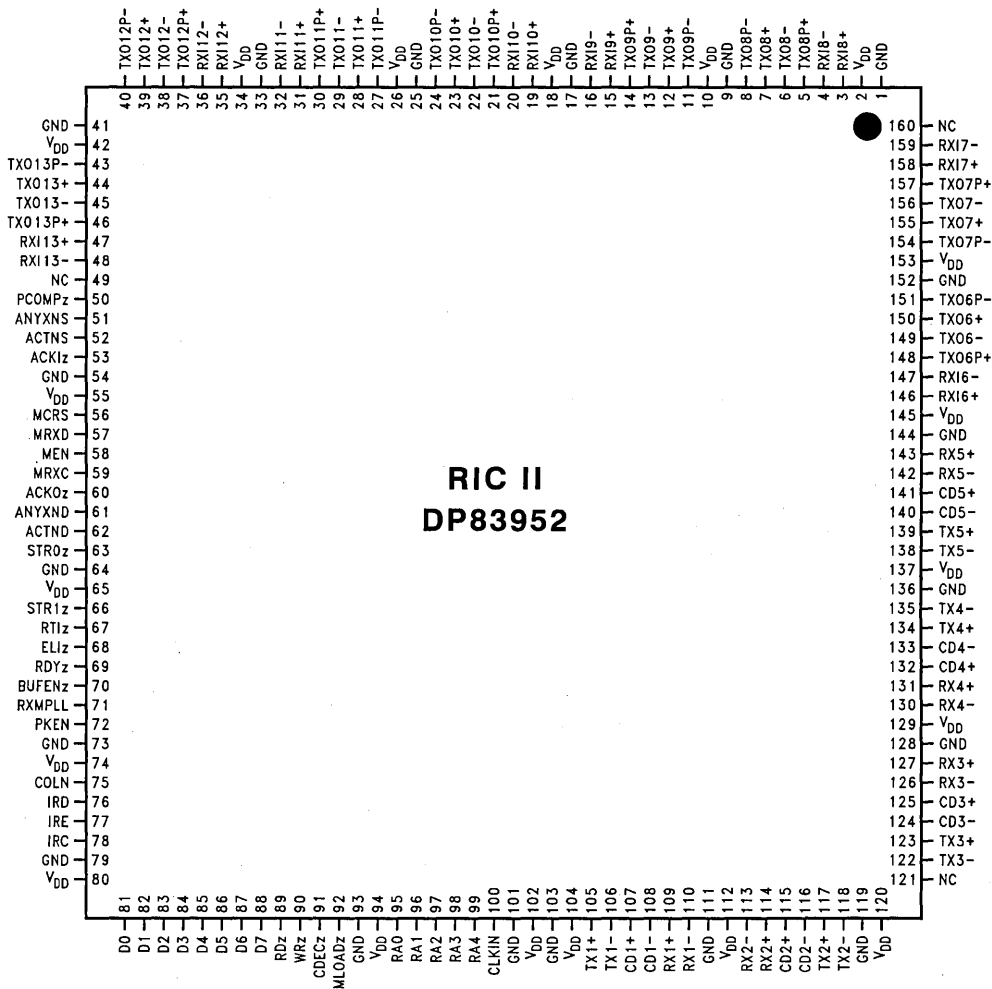
| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | 80      |
| GND             | 79      |
| IRC             | 78      |
| IRE             | 77      |
| IRD             | 76      |
| COLN            | 75      |
| V <sub>CC</sub> | 74      |
| GND             | 73      |
| PKEN            | 72      |
| RXMPLL          | 71      |
| BUFEN           | 70      |
| RDY             | 69      |
| ELI             | 68      |
| RTI             | 67      |
| STR1            | 66      |
| V <sub>CC</sub> | 65      |
| GND             | 64      |
| STR0            | 63      |
| ACTND           | 62      |
| ANYXND          | 61      |
| ACK0            | 60      |
| MRXC            | 59      |
| MEN             | 58      |
| MRXD            | 57      |
| MCRS            | 56      |
| V <sub>CC</sub> | 55      |
| GND             | 54      |
| ACK1            | 53      |
| ACTNS           | 52      |
| ANYXNS          | 51      |
| PCOMP           | 50      |
| NC              | 49      |
| RXI13–          | 48      |
| RXI13+          | 47      |
| TXO13P+         | 46      |
| TXO13–          | 45      |
| TXO13+          | 44      |
| TXO13P–         | 43      |
| V <sub>CC</sub> | 42      |
| GND             | 41      |

| Pin Name            | Pin No. |
|---------------------|---------|
| V <sub>CC</sub>     | 120     |
| GND                 | 119     |
| TX2–                | 118     |
| TX2+                | 117     |
| CD2–                | 116     |
| CD2+                | 115     |
| RX2+                | 114     |
| RX2–                | 113     |
| V <sub>CC</sub>     | 112     |
| GND                 | 111     |
| RX1–                | 110     |
| RX1+                | 109     |
| CD1–                | 108     |
| CD1+                | 107     |
| TX1–                | 106     |
| TX1+                | 105     |
| V <sub>CC</sub>     | 104     |
| GND                 | 103     |
| V <sub>CC</sub>     | 102     |
| GND                 | 101     |
| CLKIN               | 100     |
| PA4                 | 99      |
| PA3                 | 98      |
| PA2                 | 97      |
| PA1                 | 96      |
| PA0                 | 95      |
| V <sub>CC</sub> PLL | 94      |
| GNDPLL              | 93      |
| MLOAD               | 92      |
| CDEC                | 91      |
| WR                  | 90      |
| RD                  | 89      |
| D7                  | 88      |
| D6                  | 87      |
| D5                  | 86      |
| D4                  | 85      |
| D3                  | 84      |
| D2                  | 83      |
| D1                  | 82      |
| D0                  | 81      |

| Pin Name        | Pin No. |
|-----------------|---------|
| NC              | 160     |
| RXI7–           | 159     |
| RXI7+           | 158     |
| TXO7P+          | 157     |
| TXO7–           | 156     |
| TXO7+           | 155     |
| TXO7P–          | 154     |
| V <sub>CC</sub> | 153     |
| GND             | 152     |
| TXO6P–          | 151     |
| TXO6+           | 150     |
| TXO6–           | 149     |
| TXO6P+          | 148     |
| RXI6–           | 147     |
| RXI6+           | 146     |
| V <sub>CC</sub> | 145     |
| GND             | 144     |
| RX5+            | 143     |
| RX5–            | 142     |
| CD5+            | 141     |
| CD5–            | 140     |
| TX5+            | 139     |
| TX5–            | 138     |
| V <sub>CC</sub> | 137     |
| GND             | 136     |
| TX4–            | 135     |
| TX4+            | 134     |
| CD4–            | 133     |
| CD4+            | 132     |
| RX4+            | 131     |
| RX4–            | 130     |
| V <sub>CC</sub> | 129     |
| GND             | 128     |
| RX3+            | 127     |
| RX3–            | 126     |
| CD3+            | 125     |
| CD3–            | 124     |
| TX3+            | 123     |
| TX3–            | 122     |
| NC              | 121     |

Note: NC = No Connect

2.0 Connection Diagrams (Continued)



Ports 6-13 TP  
Ports 1-5 AUI

TL/F/12499-3

## 2.0 Connection Diagrams (Continued)

### Pin Table (1-7 AUI + 8-13 T.P. Ports)

| Pin Name        | Pin No. |
|-----------------|---------|
| TXO12P-         | 40      |
| TXO12+          | 39      |
| TXO12-          | 38      |
| TXO12P+         | 37      |
| RXI12-          | 36      |
| RXI12+          | 35      |
| V <sub>CC</sub> | 34      |
| GND             | 33      |
| RXI11-          | 32      |
| RXI11+          | 31      |
| TXO11P+         | 30      |
| TXO11-          | 29      |
| TXO11+          | 28      |
| TXO11P-         | 27      |
| V <sub>CC</sub> | 26      |
| GND             | 25      |
| TXO10P-         | 24      |
| TXO10+          | 23      |
| TXO10-          | 22      |
| TXO10P+         | 21      |
| RXI10-          | 20      |
| RXI10+          | 19      |
| V <sub>CC</sub> | 18      |
| GND             | 17      |
| RXI9-           | 16      |
| RXI9+           | 15      |
| TXO9P+          | 14      |
| TXO9-           | 13      |
| TXO9+           | 12      |
| TXO9P-          | 11      |
| V <sub>CC</sub> | 10      |
| GND             | 9       |
| TXO8P-          | 8       |
| TXO8+           | 7       |
| TXO8-           | 6       |
| TXO8P+          | 5       |
| RXI8-           | 4       |
| RXI8+           | 3       |
| V <sub>CC</sub> | 2       |
| GND             | 1       |

| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | 80      |
| GND             | 79      |
| IRC             | 78      |
| IRE             | 77      |
| IRD             | 76      |
| COLN            | 75      |
| V <sub>CC</sub> | 74      |
| GND             | 73      |
| PKEN            | 72      |
| RXMPLL          | 71      |
| BUFEN           | 70      |
| RDY             | 69      |
| ELI             | 68      |
| RTI             | 67      |
| STR1            | 66      |
| V <sub>CC</sub> | 65      |
| GND             | 64      |
| STR0            | 63      |
| ACTND           | 62      |
| ANYXND          | 61      |
| ACK0            | 60      |
| MRXC            | 59      |
| MEN             | 58      |
| MRXD            | 57      |
| MCRS            | 56      |
| V <sub>CC</sub> | 55      |
| GND             | 54      |
| ACK1            | 53      |
| ACTNS           | 52      |
| ANYXNS          | 51      |
| PCOMP           | 50      |
| NC              | 49      |
| RXI13-          | 48      |
| RXI13+          | 47      |
| TXO13P+         | 46      |
| TXO13-          | 45      |
| TXO13+          | 44      |
| TXO13P-         | 43      |
| V <sub>CC</sub> | 42      |
| GND             | 41      |

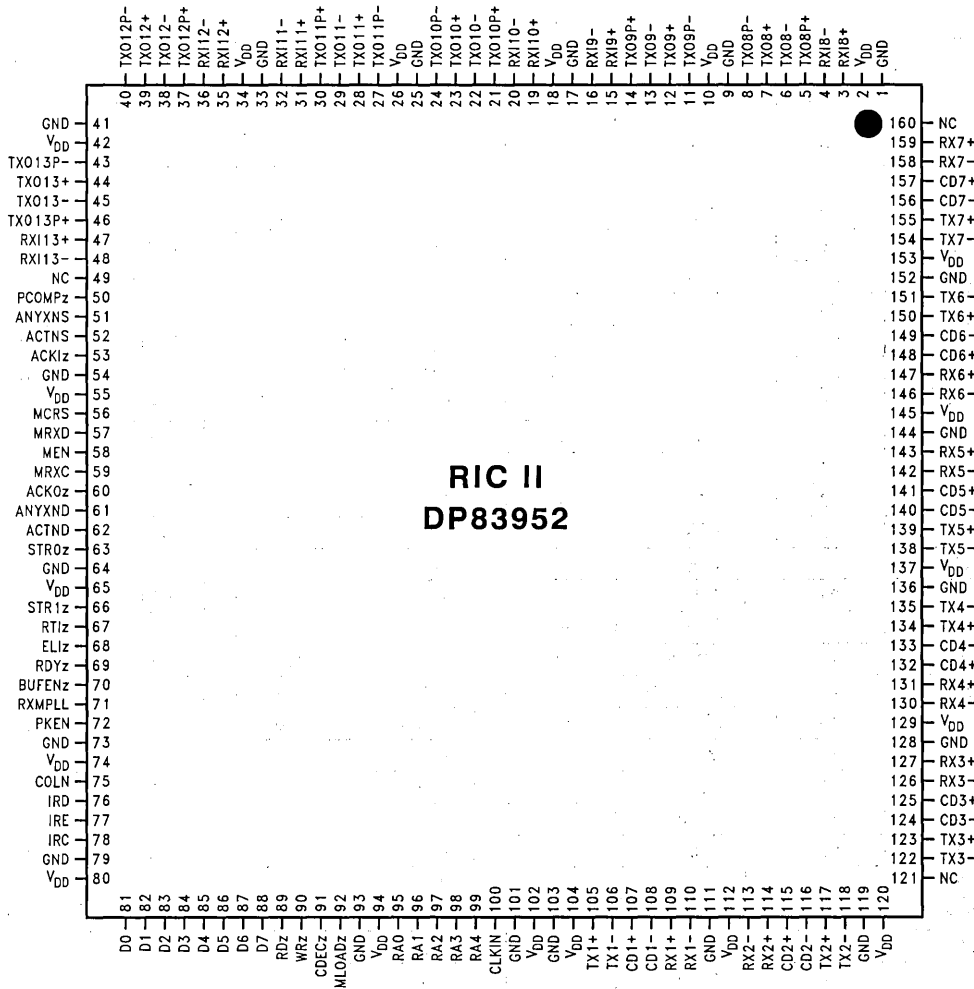
| Pin Name            | Pin No. |
|---------------------|---------|
| V <sub>CC</sub>     | 120     |
| GND                 | 119     |
| TX2-                | 118     |
| TX2+                | 117     |
| CD2-                | 116     |
| CD2+                | 115     |
| RX2+                | 114     |
| RX2-                | 113     |
| V <sub>CC</sub>     | 112     |
| GND                 | 111     |
| RX1-                | 110     |
| RX1+                | 109     |
| CD1-                | 108     |
| CD1+                | 107     |
| TX1-                | 106     |
| TX1+                | 105     |
| V <sub>CC</sub>     | 104     |
| GND                 | 103     |
| V <sub>CC</sub>     | 102     |
| GND                 | 101     |
| CLKIN               | 100     |
| PA4                 | 99      |
| PA3                 | 98      |
| PA2                 | 97      |
| PA1                 | 96      |
| PA0                 | 95      |
| V <sub>CC</sub> PLL | 94      |
| GNDPLL              | 93      |
| MLOAD               | 92      |
| CDEC                | 91      |
| WR                  | 90      |
| RD                  | 89      |
| D7                  | 88      |
| D6                  | 87      |
| D5                  | 86      |
| D4                  | 85      |
| D3                  | 84      |
| D2                  | 83      |
| D1                  | 82      |
| D0                  | 81      |

| Pin Name        | Pin No. |
|-----------------|---------|
| NC              | 160     |
| RX7+            | 159     |
| RX7-            | 158     |
| CD7+            | 157     |
| CD7-            | 156     |
| TX7+            | 155     |
| TX7-            | 154     |
| V <sub>CC</sub> | 153     |
| GND             | 152     |
| TX6-            | 151     |
| TX6+            | 150     |
| CD6-            | 149     |
| CD6+            | 148     |
| RX6+            | 147     |
| RX6-            | 146     |
| V <sub>CC</sub> | 145     |
| GND             | 144     |
| RX5+            | 143     |
| RX5-            | 142     |
| CD5+            | 141     |
| CD5-            | 140     |
| TX5+            | 139     |
| TX5-            | 138     |
| V <sub>CC</sub> | 137     |
| GND             | 136     |
| TX4-            | 135     |
| TX4+            | 134     |
| CD4-            | 133     |
| CD4+            | 132     |
| RX4+            | 131     |
| RX4-            | 130     |
| V <sub>CC</sub> | 129     |
| GND             | 128     |
| RX3+            | 127     |
| RX3-            | 126     |
| CD3+            | 125     |
| CD3-            | 124     |
| TX3+            | 123     |
| TX3-            | 122     |
| NC              | 121     |

Note: NC = No Connect



2.0 Connection Diagrams (Continued)



Ports 8-13 TP  
Ports 1-7 AUI

TL/F/12499-4

## 2.0 Connection Diagrams (Continued)

Pin Table (All AUI Ports)

| Pin Name        | Pin No. |
|-----------------|---------|
| TX12-           | 40      |
| TX12+           | 39      |
| CD12-           | 38      |
| CD12+           | 37      |
| RX12+           | 36      |
| RX12-           | 35      |
| V <sub>CC</sub> | 34      |
| GND             | 33      |
| RX11+           | 32      |
| RX11-           | 31      |
| CD11+           | 30      |
| CD11-           | 29      |
| TX11+           | 28      |
| TX11-           | 27      |
| V <sub>CC</sub> | 26      |
| GND             | 25      |
| TX10-           | 24      |
| TX10+           | 23      |
| CD10-           | 22      |
| CD10+           | 21      |
| RX10+           | 20      |
| RX10-           | 19      |
| V <sub>CC</sub> | 18      |
| GND             | 17      |
| RX9+            | 16      |
| RX9-            | 15      |
| CD9+            | 14      |
| CD9-            | 13      |
| TX9+            | 12      |
| TX9-            | 11      |
| V <sub>CC</sub> | 10      |
| GND             | 9       |
| TX8-            | 8       |
| TX8+            | 7       |
| CD8-            | 6       |
| CD8+            | 5       |
| RX8+            | 4       |
| RX8-            | 3       |
| V <sub>CC</sub> | 2       |
| GND             | 1       |

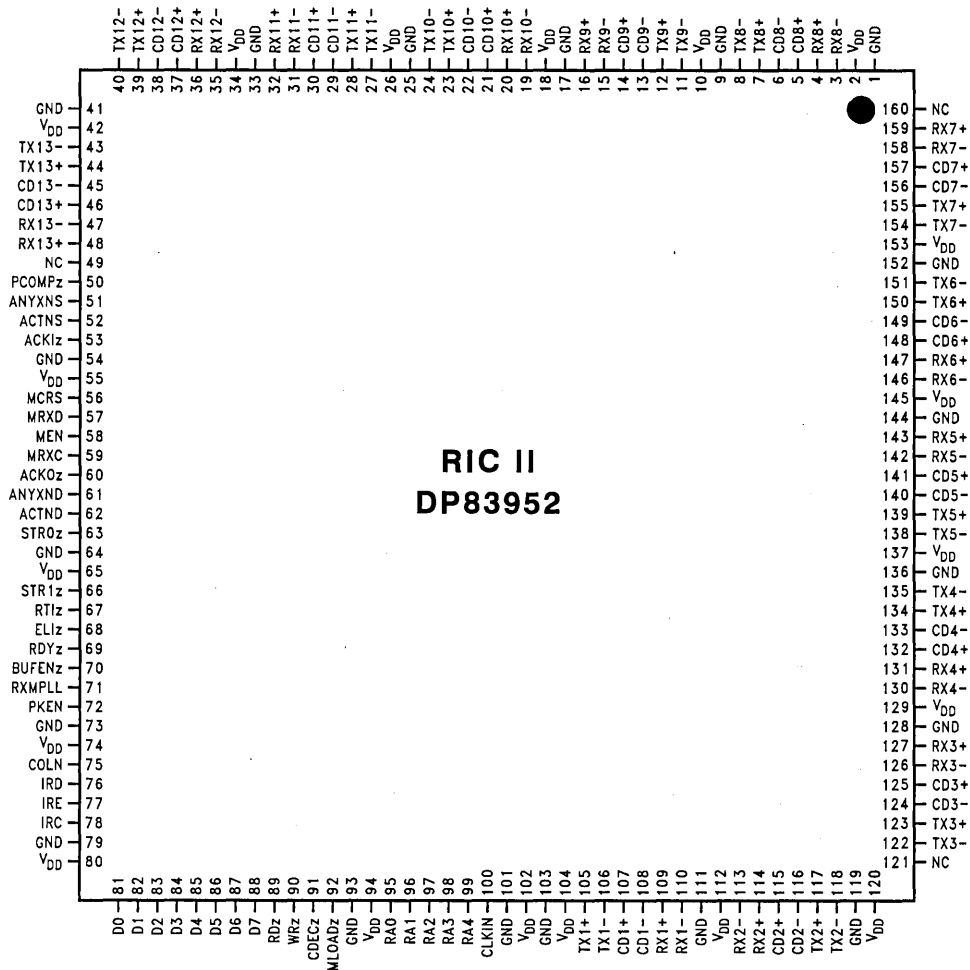
| Pin Name        | Pin No. |
|-----------------|---------|
| V <sub>CC</sub> | 80      |
| GND             | 79      |
| IRC             | 78      |
| IRE             | 77      |
| IRD             | 76      |
| COLN            | 75      |
| V <sub>CC</sub> | 74      |
| GND             | 73      |
| PKEN            | 72      |
| RXMPLL          | 71      |
| BUFEN           | 70      |
| RDY             | 69      |
| ELI             | 68      |
| RTI             | 67      |
| STR1            | 66      |
| V <sub>CC</sub> | 65      |
| GND             | 64      |
| STR0            | 63      |
| ACTND           | 62      |
| ANYXND          | 61      |
| ACK0            | 60      |
| MRXC            | 59      |
| MEN             | 58      |
| MRXD            | 57      |
| MCRS            | 56      |
| V <sub>CC</sub> | 55      |
| GND             | 54      |
| ACK1            | 53      |
| ACTNS           | 52      |
| ANYXNS          | 51      |
| PCOMP           | 50      |
| NC              | 49      |
| RX13+           | 48      |
| RX13-           | 47      |
| CD13+           | 46      |
| CD13-           | 45      |
| TX13+           | 44      |
| TX13-           | 43      |
| V <sub>CC</sub> | 42      |
| GND             | 41      |

| Pin Name            | Pin No. |
|---------------------|---------|
| V <sub>CC</sub>     | 120     |
| GND                 | 119     |
| TX2-                | 118     |
| TX2+                | 117     |
| CD2-                | 116     |
| CD2+                | 115     |
| RX2+                | 114     |
| RX2-                | 113     |
| V <sub>CC</sub>     | 112     |
| GND                 | 111     |
| RX1-                | 110     |
| RX1+                | 109     |
| CD1-                | 108     |
| CD1+                | 107     |
| TX1-                | 106     |
| TX1+                | 105     |
| V <sub>CC</sub>     | 104     |
| GND                 | 103     |
| V <sub>CC</sub>     | 102     |
| GND                 | 101     |
| CLKIN               | 100     |
| PA4                 | 99      |
| PA3                 | 98      |
| PA2                 | 97      |
| PA1                 | 96      |
| PA0                 | 95      |
| V <sub>CC</sub> PLL | 94      |
| GNDPLL              | 93      |
| MLOAD               | 92      |
| CDEC                | 91      |
| WR                  | 90      |
| RD                  | 89      |
| D7                  | 88      |
| D6                  | 87      |
| D5                  | 86      |
| D4                  | 85      |
| D3                  | 84      |
| D2                  | 83      |
| D1                  | 82      |
| D0                  | 81      |

| Pin Name        | Pin No. |
|-----------------|---------|
| NC              | 160     |
| RX7+            | 159     |
| RX7-            | 158     |
| CD7+            | 157     |
| CD7-            | 156     |
| TX7+            | 155     |
| TX7-            | 154     |
| V <sub>CC</sub> | 153     |
| GND             | 152     |
| TX6-            | 151     |
| TX6+            | 150     |
| CD6-            | 149     |
| CD6+            | 148     |
| RX6+            | 147     |
| RX6-            | 146     |
| V <sub>CC</sub> | 145     |
| GND             | 144     |
| RX5+            | 143     |
| RX5-            | 142     |
| CD5+            | 141     |
| CD5-            | 140     |
| TX5+            | 139     |
| TX5-            | 138     |
| V <sub>CC</sub> | 137     |
| GND             | 136     |
| TX4-            | 135     |
| TX4+            | 134     |
| CD4-            | 133     |
| CD4+            | 132     |
| RX4+            | 131     |
| RX4-            | 130     |
| V <sub>CC</sub> | 129     |
| GND             | 128     |
| RX3+            | 127     |
| RX3-            | 126     |
| CD3+            | 125     |
| CD3-            | 124     |
| TX3+            | 123     |
| TX3-            | 122     |
| NC              | 121     |

Note: NC = No Connect

### 2.0 Connection Diagrams (Continued)



All AUI Ports

TL/F/12499-5

### 3.0 Pin Description

| Pin Name                                                                                          | Driver Type | I/O | Description                                        |
|---------------------------------------------------------------------------------------------------|-------------|-----|----------------------------------------------------|
| <b>NETWORK INTERFACE PINS (On-Chip Transceiver Mode)</b>                                          |             |     |                                                    |
| RX12- to RX113-                                                                                   | TP          | I   | Twisted Pair Receive Input Negative                |
| RX12+ to RX113+                                                                                   | TP          | I   | Twisted Pair Receive Input Positive                |
| TXOP2- to TXOP13-                                                                                 | TT          | O   | Twisted Pair Pre-Emphasis Transmit Output Negative |
| TXO2- to TXO13-                                                                                   | TT          | O   | Twisted Pair Transmit Output Negative              |
| TXO2+ to TXO13+                                                                                   | TT          | O   | Twisted Pair Transmit Output Positive              |
| TXOP2+ to TXOP13+                                                                                 | TT          | O   | Twisted Pair Pre-Emphasis Transmit Output Positive |
| CD1+                                                                                              | AL          | I   | AUI Collision Detect Input Positive                |
| CD1-                                                                                              | AL          | I   | AUI Collision Detect Input Negative                |
| RX1+                                                                                              | AL          | I   | AUI Receive Input Positive                         |
| RX1-                                                                                              | AL          | I   | AUI Receive Input Negative                         |
| TX1+                                                                                              | AD          | O   | AUI Transmit Output Positive                       |
| TX1-                                                                                              | AD          | O   | AUI Transmit Output Negative                       |
| <b>NETWORK INTERFACE PINS (External Transceiver Mode AUI Signal Level Compatibility Selected)</b> |             |     |                                                    |
| TX2+ to TX13+                                                                                     | AL          | O   | Transmit Output Positive                           |
| TX2- to TX13-                                                                                     | AL          | O   | Transmit Output Negative                           |
| CD2+ to CD13+                                                                                     | AL          | I   | Collision Input Positive                           |
| CD2- to CD13-                                                                                     | AL          | I   | Collision Input Negative                           |
| RX2+ to RX13+                                                                                     | AL          | I   | Receive Input Positive                             |
| RX2- to RX13-                                                                                     | AL          | I   | Receive Input Negative                             |
| CD1+                                                                                              | AL          | I   | AUI Collision Detect Input Positive                |
| CD1-                                                                                              | AL          | I   | AUI Collision Detect Input Negative                |
| RX1+                                                                                              | AL          | I   | AUI Receive Input Positive                         |
| RX1-                                                                                              | AL          | I   | AUI Receive Input Negative                         |
| TX1+                                                                                              | AD          | O   | AUI Transmit Output Positive                       |
| TX1-                                                                                              | AD          | O   | AUI Transmit Output Negative                       |

**Note:** AD = AUI level and Drive compatible, TP = Twisted Pair interface compatible, AL = AUI Level compatible, TT = TTL compatible, I = Input, O = Output, B = Bi-directional, Z = TRI-STATE®, C = CMOS compatible.

### 3.0 Pin Description (Continued)

| Pin Name                  | Driver Type | I/O  | Description                                                                                                                                                                                                                                                                                                                                                                                                                      |
|---------------------------|-------------|------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PROCESSOR BUS PINS</b> |             |      |                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| RA0-RA4                   | TT          | I    | <b>REGISTER ADDRESS INPUTS:</b> These five pins are used to select a register to be read or written. The state of these inputs are ignored when the read, write and mode load input strobes are high. (Even under these conditions these inputs must not be allowed to float to an undefined logic state).                                                                                                                       |
| $\overline{\text{STR0}}$  | C           | O    | <b>DISPLAY UPDATE STROBE 0</b><br><b>Maximum Display Mode:</b> This signal controls the latching of display data for network ports 1 to 7 into the off chip display latches.<br><b>Minimum Display Mode:</b> This signal controls the latching of display data for the RIC II into the off chip display latch.<br>During processor access cycles (read or write is asserted) this signal is inactive (high).                     |
| $\overline{\text{STR1}}$  | C           | O    | <b>DISPLAY UPDATE STROBE 1</b><br><b>Maximum Display Mode:</b> This signal controls the latching of display data for network ports 8 to 13 into the off-chip display latches.<br><b>Minimum Display Mode:</b> No operation<br>During processor access cycles (read or write is asserted) this signal is inactive (high).                                                                                                         |
| D0-D7                     | TT          | B, Z | <b>DATA BUS:</b> (Note 1)<br><b>Display Update Cycles:</b> These pins become outputs providing display data and port address information. Address information only available in Maximum Display mode.<br><b>Processor Access Cycles:</b> Data input or output is performed via these pins. The read, write and mode load inputs control the direction of the signals.                                                            |
| $\overline{\text{BUFEN}}$ | C           | O    | <b>BUFFER ENABLE:</b> This output controls the TRI-STATE operation of the bus transceiver which provides the interface between the RIC II's data pins and the processor's data bus. (Note 2)                                                                                                                                                                                                                                     |
| $\overline{\text{RDY}}$   | C           | O    | <b>DATA READY STROBE:</b> The falling edge of this signal during a read cycle indicates that data is stable and valid for sampling. In write cycles the falling edge of $\overline{\text{RDY}}$ denotes that the write data has been latched by the RIC II. Therefore data must have been available and stable for this operation to be successful.                                                                              |
| $\overline{\text{ELI}}$   | C           | O    | <b>EVENT LOGGING INTERRUPT:</b> A low level on the $\overline{\text{ELI}}$ output indicates the RIC II's hub management logic requires CPU attention. The interrupt is cleared by accessing the Port Event Recording register or Event Counter that produced it. All interrupt sources may be masked.                                                                                                                            |
| $\overline{\text{RTI}}$   | C           | O    | <b>REAL TIME INTERRUPT:</b> A low level on the $\overline{\text{RTI}}$ output indicates the RIC II's real time (packet specific) interrupt logic requires CPU attention. The interrupt is cleared by reading the Real Time Interrupt Status register. All interrupt sources may be masked.                                                                                                                                       |
| $\overline{\text{CDEC}}$  | TT          | I    | <b>COUNTER DECREMENT:</b> A rising edge on the $\overline{\text{CDEC}}$ input strobe decrements all of the RIC II's Port Event Counters by one. This input is internally synchronized and if necessary the operation of the signal is delayed if there is a simultaneous internally generated counting operation.                                                                                                                |
| $\overline{\text{WR}}$    | TT          | I    | <b>WRITE STROBE:</b> Strobe from the CPU used to write an internal register defined by the RA0-RA4 inputs.                                                                                                                                                                                                                                                                                                                       |
| $\overline{\text{RD}}$    | TT          | I    | <b>READ STROBE:</b> Strobe from the CPU used to read an internal register defined by the RA0-RA4 inputs.                                                                                                                                                                                                                                                                                                                         |
| $\overline{\text{MLOAD}}$ | TT          | I    | <b>DEVICE RESET AND MODE LOAD:</b> When this input is low all of the RIC II's state machines, counters and network ports are reset and held inactive. On the rising edge of $\overline{\text{MLOAD}}$ the logic levels present on the D0-7 pins and RA0-RA4 inputs are latched into the RIC II's configuration registers. The rising edge of $\overline{\text{MLOAD}}$ also signals the beginning of the display test operation. |

**Note 1:** The data pins remain in their display update function, i.e., asserted as outputs unless either the read or write strobe is asserted.

**Note 2:** The buffer enable output indicates the function of the data pins. When it is high they are performing display update cycles, when it is low a processor access or mode load cycle is occurring.

### 3.0 Pin Description (Continued)

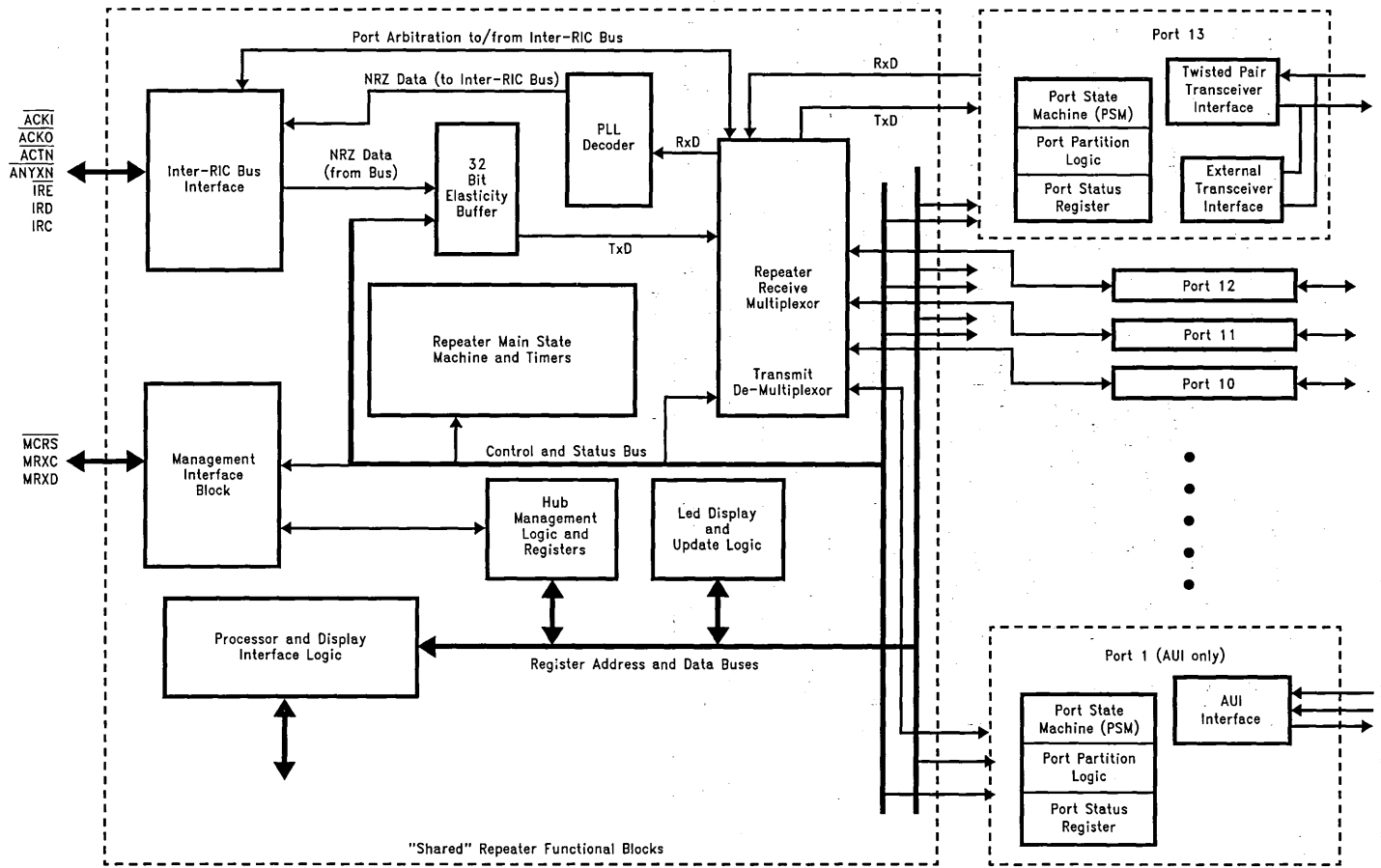
| Pin Name                  | Driver Type | I/O  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------------|-------------|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>INTER-RIC BUS PINS</b> |             |      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| $\overline{\text{ACKI}}$  | TT          | I    | <b>ACKNOWLEDGE INPUT:</b> Input to the network ports' arbitration chain.                                                                                                                                                                                                                                                                                                                                                                                              |
| $\overline{\text{ACKO}}$  | TT          | O    | <b>ACKNOWLEDGE OUTPUT:</b> Output from the network ports' arbitration chain.                                                                                                                                                                                                                                                                                                                                                                                          |
| IRD                       | TT          | B, Z | <b>INTER-RIC DATA:</b> When asserted as an output this signal provides a serial data stream in NRZ format. The signal is asserted by a RIC II when it is receiving data from one of its network segments. The default condition of this signal is to be an input. In this state it may be driven by other devices on the Inter-RIC bus.                                                                                                                               |
| IRE                       | TT          | B, Z | <b>INTER-RIC ENABLE:</b> When asserted as an output this signal provides an activity framing enable for the serial data stream. The signal is asserted by a RIC II when it is receiving data from one of its network segments. The default condition of this signal is to be an input. In this state it may be driven by other devices on the Inter-RIC bus.                                                                                                          |
| IRC                       | TT          | B, Z | <b>INTER-RIC CLOCK:</b> When asserted as an output this signal provides a clock signal for the serial data stream. Data (IRD) is changed on the falling edge of the clock. The signal is asserted by a RIC II when it is receiving data from one of its network segments. The default condition of this signal is to be an input. When an input IRD is sampled on the rising edge of the clock. In this state it may be driven by other devices on the Inter-RIC bus. |
| COLN                      | TT          | B, Z | <b>COLLISION ON PORT N:</b> This denotes that a collision is occurring on the port receiving the data packet. The default condition of this signal is to be an input. In this state it may be driven by other devices on the Inter-RIC bus.                                                                                                                                                                                                                           |
| PKEN                      | C           | O    | <b>PACKET ENABLE:</b> This output acts as an active high enable for an external bus transceiver (if required) for the IRE, IRC, IRD and COLN signals. When high the bus transceiver should be transmitting on to the bus, i.e. this RIC II is driving the IRD, IRE, IRC, and COLN bus lines. When low the bus transceiver should receive from the bus.                                                                                                                |
| CLKIN                     | TT          | I    | <b>40 MHz CLOCK INPUT:</b> This input is used to generate the RIC II's timing reference for the state machines, and phase lock loop decoder.                                                                                                                                                                                                                                                                                                                          |
| ACTND                     | OD          | O    | <b>ACTIVITY ON PORT N DRIVE:</b> This output is active when the RIC II is receiving data or collision information from one of its network segments.                                                                                                                                                                                                                                                                                                                   |
| ACTNS                     | TT          | I    | <b>ACTIVITY ON PORT N SENSE:</b> This input senses when this or another RIC II in a multi-RIC II system is receiving data or collision information.                                                                                                                                                                                                                                                                                                                   |
| ANYXND                    | OD          | O    | <b>ACTIVITY ON ANY PORT EXCLUDING PORT N DRIVE:</b> This output is active when a RIC II is experiencing a transmit collision or multiple ports have active collisions on their network segments.                                                                                                                                                                                                                                                                      |
| ANYXNS                    | TT          | I    | <b>ACTIVITY ON ANY PORT EXCLUDING PORT N SENSE:</b> This input senses when this RIC II or other RIC IIs in a multi-RIC II system are experiencing transmit collisions or multiple ports have active collisions on their network segments.                                                                                                                                                                                                                             |

TT = TTL compatible, B = Bi-directional, C = CMOS compatible, OD = Open Drain, I = Input, O = Output

### 3.0 Pin Description (Continued)

| Pin Name                     | Driver Type | I/O  | Description                                                                                                                                                                                                                                                                                                    |
|------------------------------|-------------|------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>MANAGEMENT BUS PINS</b>   |             |      |                                                                                                                                                                                                                                                                                                                |
| MRXC                         | TT          | O, Z | <b>MANAGEMENT RECEIVE CLOCK:</b> When asserted this signal provides a clock signal for the MRXD serial data stream. The MRXD signal is changed on the falling edge of this clock. The signal is asserted when a RIC II is receiving data from one of its network segments. Otherwise the signal is inactive.   |
| MCRS                         | TT          | B, Z | <b>MANAGEMENT CARRIER SENSE:</b> When asserted this signal provides an activity framing enable for the serial data stream. The signal is asserted when a RIC II is receiving data from one of its network segments. Otherwise the signal is an input.                                                          |
| MRXD                         | TT          | O, Z | <b>MANAGEMENT RECEIVE DATA:</b> When asserted this signal provides a serial data stream in NRZ format. The data stream is made up of the data packet and RIC II status information. The signal is asserted when a RIC II is receiving data from one of its network segments. Otherwise the signal is inactive. |
| MEN                          | C           | O    | <b>MANAGEMENT BUS OUTPUT ENABLE:</b> This output acts as an active high enable for an external bus transceiver (if required) for the MRXC, MCRS and MRXD signals. When high the bus transceiver should be transmitting on to the bus.                                                                          |
| PCOMP                        | TT          | I    | <b>PACKET COMPRESS:</b> This input is used to activate the RIC II's packet compress logic. A low level on this signal when MCRS is active will cause that packet to be compressed. If PCOMP is tied low all packets are compressed, if PCOMP is tied high packet compression is inhibited.                     |
| <b>POWER AND GROUND PINS</b> |             |      |                                                                                                                                                                                                                                                                                                                |
| V <sub>CC</sub>              |             |      | Positive Supply                                                                                                                                                                                                                                                                                                |
| GND                          |             |      | Negative Supply                                                                                                                                                                                                                                                                                                |
| <b>EXTERNAL DECODER PINS</b> |             |      |                                                                                                                                                                                                                                                                                                                |
| RXM                          | TT          | O    | <b>RECEIVE DATA MANCHESTER FORMAT:</b> This output makes the data, in Manchester format, received by port N available for test purposes. If not used for testing this pin should be left open.                                                                                                                 |

TT = TTL compatible, B = Bi-directional, C = CMOS compatible, OD = Open Drain, I = Input, O = Output



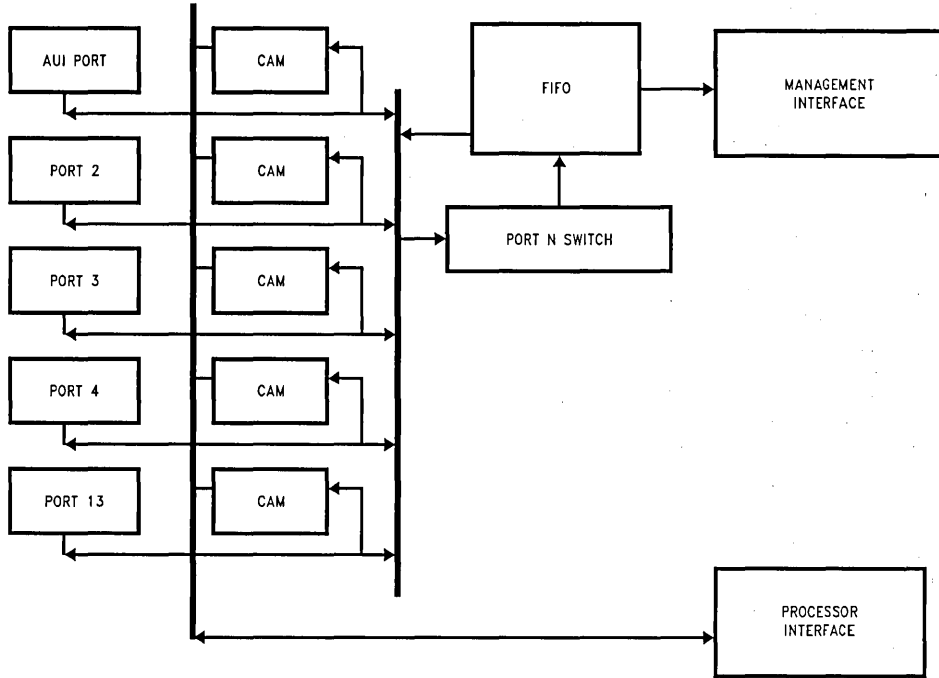
3-96

FIGURE 4-1



4.0 Block Diagram (Continued)

RIC II Port Architecture Block Diagram



TL/F/12499-7

**Note:** The block diagram for the RIC II, when used in the non-secure mode, is identical to the block diagram for the RIC device (*Figure 4-1*). When RIC II is used in the secure mode, the added functional blocks for the security logic are used in the operation of the device (*Figure 4-1a*).

FIGURE 4-1a

## 5.0 Functional Description

The IEEE 802.3 repeater specification details a number of functions a repeater system must perform. These requirements allied with a need for the implementation to be multi-port strongly favors the choice of a modular design style. In such a design, functionality is split between those tasks common to all data channels and those exclusive to each individual channel. The RIC II follows this approach, certain functional blocks are replicated for each network attachment, (also known as a repeater port), and others are shared.

The following sections provide an overview of the RIC II architecture. First, RIC II feature enhancements from the RIC is discussed. Then, the RIC II functional blocks are described.

### 5.1 SUMMARY OF DP83952 RIC II FEATURE ENHANCEMENTS FROM DP83950B RIC

1. Pin Compatibility. The DP83952VUL RIC II is fully pin compatible with the DP83950BVQB RIC device in the 160-pin Plastic Quad Flat Pack (PQFP) package.
2. Addition of network security. The DP83952 RIC II features significant per port security capability. As a single chip repeater, RIC II provides security using 58 internal CAM (Content Addressable Memory) locations for all 13 ports.  
Unauthorized nodes can be restricted from intruding and/or eavesdropping into the network by preventing them from having access to valid packet data. The RIC II utilizes internal CAMs to store/compare addresses of valid nodes. Complete security operation of RIC II is explained later in this datasheet.
3. Two dedicated CAM locations per port and 32 shareable CAM entries are provided to store Ethernet addresses. Both the port and the shareable CAM locations (48 bits wide) are on-chip.
4. Faster processor access for efficient data gathering.
5. Addition of thirteen 8-bit wide counters for collecting more statistics.

### 5.2 OVERVIEW OF RIC II FUNCTIONS

#### SEGMENT SPECIFIC BLOCK: NETWORK PORT

As shown in the Block Diagram, the segment specific blocks consist of:

1. One or more physical layer interfaces.
2. A logic block required for performing repeater operations upon that particular segment. This is known as the "port" logic since it is the access "port" the segment has to the rest of the network.

This function is repeated 13 times in the RIC II (one for each port) and is shown on the right side of the Block Diagram, *Figure 4-1*.

The physical layer interface depends upon the port. Port 1 has an AUI compliant interface for use with AUI compatible transceiver boxes and cable. Ports 2 to 13 may be configured for use with one of two interfaces: twisted pair or an external transceiver. The former utilizes the RIC II's on-chip 10BASE-T transceivers, the latter allows connection to external transceivers. When using the external transceiver mode the interface is AUI compatible. Although AUI com-

patible transceivers are supported, when an interface cable is used, external transceivers should always be used for the repeater system.

Inside the port logic there are 4 distinct functions:

1. The port state machine "PSM" is required to perform data and collision repetition as described by the repeater specification, for example, it determines whether this port should be receiving from or transmitting to its network segment.
2. The port partition logic implements the segment partitioning algorithm. This algorithm is defined by the IEEE specification and is used to protect the network from malfunctioning segments.
3. The port status register reflects the current status of the port. It may be accessed by a system processor to obtain this status or to perform certain port configuration operations, such as port disable.
4. The Port Security Configuration Logic determines if the packet data will be transmitted/received intact, or as pseudo random data. Two dedicated CAM locations per port are available for learning/storing/comparing port source addresses.

#### SHARED FUNCTIONAL BLOCKS: REPEATER CORE LOGIC

The shared functional blocks consists of the Repeater Main State Machine (MSM), Timers, a 32-bit Elasticity Buffer, PLL Decoder, Receive and Transmit Multiplexors, and Security Logic with 32 shareable CAM locations. These blocks perform the majority of the operations needed to fulfill the requirements of the IEEE repeater specification.

When a packet is received by a port it is sent via the Receive Multiplexor to the PLL Decoder. Notification of the data and collision status is sent to the main state machine via the receive multiplexor and collision activity status signals. This enables the main state machine to determine the source of the data to be repeated and the type of data to be transmitted, either data or jam pattern.

When a collision occurs, in accordance with IEEE repeater specifications, the transmit data will be a preamble/jam pattern consisting of a 1010 . . . bit pattern. Whenever a collision occurs, (during the preamble, the address field, the "type field", or the data field) the RIC II switches to the jam pattern immediately.

When RIC II is configured in the "non-secure" mode, the valid received data field is transmitted to all other ports (excluding the port with the received packet).

When RIC II is configured in the "secure" mode, the source and destination addresses within each packet are first checked. Based on this comparison, and the port configuration either:

1. a pseudo random bit pattern during the data field (of the packet) is transmitted to the particular port. Or,
2. the received data is transmitted intact.

The data remains intact on the Inter-RIC bus so other cascaded repeaters could compare the destination address with their local CAMs. On a valid source address mismatch, RIC II shall switch to random pattern both on the local transmitting ports and the Inter-RIC bus.

## 5.0 Functional Description (Continued)

The main state machine is associated with a series of timers. These ensure various IEEE specification times (referred to as the TW1 to TW6 times) are fulfilled.

A repeater unit is required to meet the same signal jitter performance as any receiving node attached to a network segment. Consequently, a phase locked loop Manchester decoder is required so that the packet may be decoded, and the jitter accumulated over the receiving segment recovered. The decode logic outputs data in non-return to zero (NRZ) format with an associated clock and enable. In this form, the packet is in a convenient format for transfer to other devices, such as network controllers and other RIC IIs, via the Inter-RIC bus (described later). The data may then be re-encoded into Manchester data and transmitted.

Reception and transmission via physical layer transceiver units causes a loss of bits in the preamble field of a data packet. The repeater specification requires this loss to be regenerated. To accomplish this, an elasticity buffer is employed to temporarily store bits in the data field of the packet.

The sequence of operation is as follows. Soon after the network segment receiving the data packet has been identified, the RIC II begins to transmit the packet preamble pattern (1010...) onto the other network segments. While the preamble is being transmitted, the Elasticity Buffer monitors the decoded received clock and data signals (this is done via the Inter-RIC bus as described later). When the start of frame delimiter "SFD" is detected, the received data stream is written into the elasticity buffer. Removal of data from the buffer for retransmission is not allowed until a valid length preamble pattern has been transmitted.

### Internal CAMs

For security purposes, RIC II employs two sets of CAMs (Content Addressable Memory) for address comparison: port CAMs, and shared CAMs.

#### PORT CAMs

RIC II provides two CAM locations (48 bits wide) per port for comparison. The two CAM locations hold the source address(es) for the incoming packets on that port. The addresses can be stored (CPU access), or learned (Learn mode). External processor/logic access to these two CAM locations is not advised or allowed while in learning mode, LME = 1, since the contents of the register may not be valid. Once the addresses are learned, then they are used to make source address and destination address comparisons. An address is learned when the packet is received with a valid CRC. External processor/logic access to these registers is fine while learning is not in progress, LME = 0 in the port security configuration register.

#### SHARED CAMs

RIC II provides thirty-two shareable CAM locations (48 bits wide) to store the Ethernet addresses associated with the ports. The Ethernet addresses are stored by writing to these CAM locations where the addresses could be shared among the thirteen ports. By using shared CAMs, multiple Ethernet addresses can be associated with a single port, or multiple ports can be allocated to a single Ethernet address. After the destination address of the received packet is completely buffered, RIC II will compare this address with the

stored addresses in the CAM locations. The source address is compared in a similar fashion. These shared CAM locations are only user definable, and will not be filled during the learning mode.

A CAM entry could be shared among the thirteen local ports. This is done through a 16-bit CAM Location Mask Register (CLMR). Each CAM entry has one of these CLMRs, therefore there are 32 registers for the 32 CAM entries. These registers could be accessed by a processor.

Since register access is performed on a byte basis, six write cycles must be done to program/enter the Ethernet address into the CAM. The upper 3 bits of the CAM Location Mask Register act as a pointer indicating which byte of the 6-byte address will be accessed next. This pointer will increment every time a read or write cycle is done to the CAM entry. The pointer starts at 1, indicating the least significant byte of the address.

Four additional registers are provided to validate the shared 32 CAM entries and are referred to as Shared CAM Validation Register 1-4 (SCVR 1-4, Page 9H, Address 16-19H). Each bit of these registers is mapped to one CAM location. An address in the CAM location will only be valid when a corresponding bit Address Valid (ADV bit) has been set in this register. RIC II will include only valid CAM locations for address comparison.

The contents of all CAM locations are unknown at power up. This will pose no problem since the ADV (Address Valid) bit is not set for the CAM. Therefore the comparison will not take place with the CAM contents.

#### INTER-RIC BUS INTERFACE

Using the RIC II in a repeater system allows the design to be constructed with many network attachments than can be supported by a single chip. The split of functions already described allows data packets, and collision status to be transferred between multiple RIC IIs, and at the same time the multiple RIC IIs still behave as a single logical repeater. Since all RIC IIs in a repeater system are identical and capable of performing any of the repetition operations, the failure of one RIC II will not cause the failure of the entire system. This is an important issue in large multiport repeaters.

RIC IIs communicate via a specialized interface known as the Inter-RIC bus. This allows the data packet to be transferred from the receiving RIC II to the other RIC IIs in the system. These RIC IIs then transmit the data stream to their segments. The notification of collisions occurring across the network is just as important as data transfers. The Inter-RIC bus has a set of status lines capable of conveying collision information between RIC IIs to ensure their main state machines operate in the appropriate manner.

#### LED INTERFACE AND HUB MANAGEMENT FUNCTION

Repeater systems usually possess optical displays indicating network activity and the status of specific repeater operations. The RIC II's display update block provides the system designer with a wide variety of indicators. The display updates are completely autonomous and merely require SSI logic devices to drive the display devices, usually made up of light emitting diodes, LEDs. The status display is very flexible allowing the user to choose those indicators appropriate for the specification of the equipment.

## 5.0 Functional Description (Continued)

RIC II has been designed with special awareness for system designers implementing large repeaters possessing hub management capabilities. Hub management uses the unique position of repeaters in a network to gather statistics about the network segments they are attached to. The RIC II provides hub management statistical data in 3 steps. Important events are gathered by the management block from logic blocks throughout the chip. These events may then be stored in on-chip latches, or counted in on-chip counters according to user supplied latching and counting masks.

The fundamental task of a hub management system implementation is to associate the current packet and any management status information with the network segment, such as the repeater port where the packet was received. The ideal system would place this combined data packet and status field in system memory for examination by hub management software. The ultimate function of the RIC II's hub management support logic is to provide this function.

To accomplish this, the RIC II utilizes a dedicated hub management interface. This is similar to the Inter-RIC bus since it allows the data packet to be recovered from the receiving RIC II. Unlike the Inter-RIC bus, the intended recipient is not another RIC II, but National Semiconductor's DP83932 "SONIC™" Network controller or the DP83957 RIB. The use of a dedicated bus allows a management status field to be appended at the end of the data packet. This can be done without affecting the operation of the repeater system.

The RIC II adds 13 more (8-bit wide) counters in addition to counters provided on the RIC DP83950B. These counters will count events specified in the Event Count and Interrupt Mask Register2 (ECIMR2) such as Frame Check Sequence, Frame Alignment Error, Partition, and Out of Window Collision. This register also includes "Reset On Read" and "Freeze When Full" control bits.

It should be noted that Counter Decrement (CDEC) will not be used with the ECIMR2. Also no real time or event logging interrupt, RTI or ELI, will be generated for this register.

### PROCESSOR INTERFACE

The RIC II's processor interface allows connection to a system processor. Data transfer occurs via an octal bi-directional data bus. The RIC II has a number of on-chip registers indicating the status of the hub management functions, chip configuration and port status. These may be accessed by providing the chosen address at the Register Address (RA4-RA0) input pins.

Display update cycles and processor accesses occur utilizing the same data bus. An on-chip arbiter in the processor/display block schedules and controls the accesses and ensures the correct information is written into the display latches. During the display update cycles the RIC II behaves as a master of its data bus. This is the default state of the data bus. Consequently, a TRI-STATE buffer must be placed between the RIC II and the system processor's data bus. This ensures bus contention is avoided during simultaneous display update cycles and processor accesses of other devices on the system bus. When the processor accesses a RIC II register, the RIC II enables the data buffer and selects the operation, either input or output, of the data pins.

For faster register accesses, RIC II provides the added feature of disabling the display update cycles. In the Lower Event Information register (Page 1H, Address 1FH) setting the Disable LED Update bit, DLU, stops the RIC LED updates. This causes the data bus to be no longer shared, therefore, RIC II is always in a slave access mode. In this mode, the maximum read/write cycle time is reduced to approximately 400 ns.

### 5.3 DESCRIPTION OF REPEATER OPERATIONS

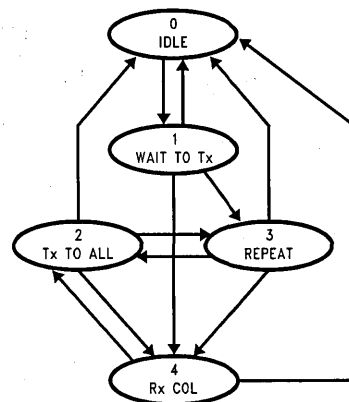
In order to implement a multi-chip repeater system which behaves as though it were a single logical repeater, special consideration must be paid to the data path used in packet repetition. For example, where in the data path are specific operations such as Manchester decoding and elasticity buffering performed. Also the system's state machines which utilize available network activity signals, must be able to accommodate the various packet repetition and collision scenarios detailed in the repeater specification.

The RIC II contains two types of inter-acting state machines. These are:

1. Port State Machines (PSMs). Every network attachment has its own PSM.
2. Main State Machine (MSM). This state machine controls the shared functional blocks as shown in the block diagram *Figure 4-1*.

### REPEATER PORT AND MAIN STATE MACHINES

These two state machines are described in the following sections. Reference is made to expressions used in the IEEE Repeater specification. For the precise definition of these terms please refer to the specification. To avoid confusion with the RIC II's implementation, where references are made to repeater states or terms as described in the IEEE specification, these items are written in *italics*. The IEEE state diagram is shown in *Figure 5-2*, the Inter-RIC bus state diagram is shown in *Figure 5-1*.



TL/F/12499-B

FIGURE 5-1. Inter-RIC Bus State Diagram

### 5.0 Functional Description (Continued)

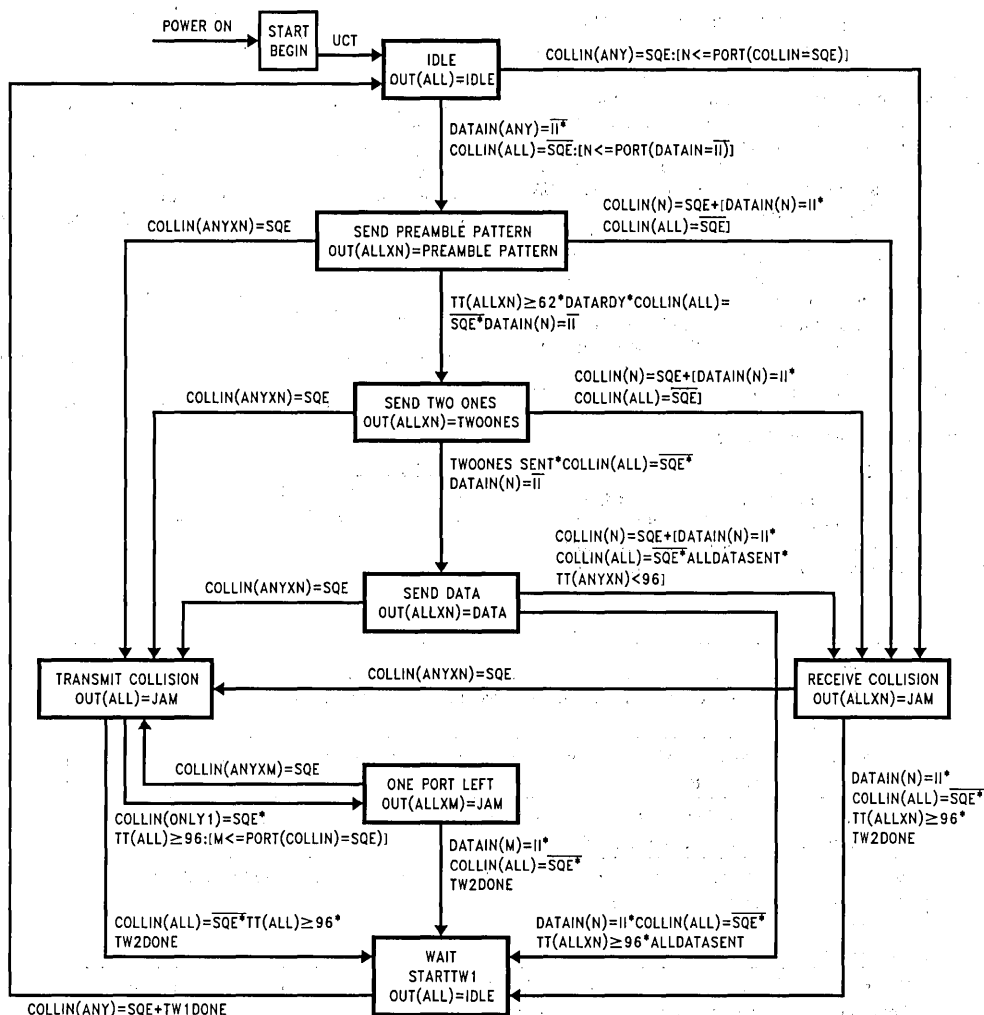


FIGURE 5-2. IEEE Repeater Main State Diagram

TL/F/12499-9

## 5.0 Functional Description (Continued)

### PORT STATE MACHINE (PSM)

There are two primary functions for the PSM as follows:

1. Control the transmission of repeated data, pseudo random data, and jam signals over the attached segment.
2. Decide whether a port will be the source of data or collision information which will be repeated over the network. This repeater port is known as *PORT N*. An arbitration process is required to enable the repeater to transition from the *IDLE* state to the *SEND PREAMBLE PATTERN* or *RECEIVE COLLISION* states, see *Figure 5-2*. This process is used to locate the port which will be *PORT N* for that particular packet. The data received from this port is directed to the PLL decoder and transmitted over the Inter-RIC bus. If the repeater enters the *TRANSMIT COLLISION* state a further arbitration operation is performed to determine which port is *PORT M*. *PORT M* is differentiated from the repeater's other ports if the repeater enters the *ONE PORT LEFT* state. In this state *PORT M* does not transmit to its segment; where as all other ports are still required to transmit to their segments.

### MAIN STATE MACHINE (MSM)

The MSM controls the operation of the shared functional blocks in each RIC II as shown in the block diagram, *Figure 4-1*, and it performs the majority of the data and collision propagation operations as defined by the IEEE specification.

The interaction of the main and port state machines is visible, in part, by observing the Inter-RIC bus.

### INTER-RIC BUS OPERATION

#### OVERVIEW

The Inter-RIC Bus consists of eight signals. These signals implement a protocol which may be used to connect multiple RIC IIs together. In this configuration, the logical function of a single repeater is maintained. The resulting multi-RIC II system is compliant to the IEEE 802.3 repeater specification and may connect several hundred network segments. An example of a multi-RIC II system is shown in *Figure 5-3*.

The Inter-RIC Bus connects multiple RIC IIs to realize the following operations:

- Port N* Identification (which port the repeater receives data from)
- Port M* Identification (which port is the last one experiencing a collision)
- Data Transfer
- RECEIVE COLLISION* identification
- TRANSMIT COLLISION* identification
- DISABLE OUTPUT* (jabber protection)

Repeater Functions

| Function                          | Action                                                                                                                              |
|-----------------------------------|-------------------------------------------------------------------------------------------------------------------------------------|
| Preamble Regeneration             | Restore the length of the preamble pattern to the defined size.                                                                     |
| Fragment Extension                | Extend received data or collision fragments to meet the minimum fragment length of 96 bits.                                         |
| Elasticity Buffer Control         | A portion of the received packet may require storage in an Elasticity Buffer to accommodate preamble regeneration.                  |
| Jam/Preamble Pattern Generation   | In cases of receive or transmit collisions, a RIC II is required to transmit a jam pattern (1010 . . .). (Note)                     |
| Transmit Collision Enforcement    | Once the <i>TRANSMIT COLLISION</i> state is entered a repeater is required to stay in this state for at least 96 network bit times. |
| Data Encoding Control             | NRZ format data from the elasticity buffer must be encoded into Manchester format data prior to retransmission.                     |
| <i>T<sub>w1</sub></i> Enforcement | Enforce the Transmit Recovery Time specification.                                                                                   |
| <i>T<sub>w2</sub></i> Enforcement | Enforce Carrier Recovery Time specification on all ports with active collisions.                                                    |

Note: This pattern is the same as that used for preamble regeneration.

## 5.0 Functional Description (Continued)

The following table briefly describes the operation of the Inter-RIC bus signals, the conditions required for a RIC II to assert a signal, and which RIC IIs (in a multi-RIC II system) would monitor the signal.

| Inter-RIC Bus Signal | Function                                                                                                                 | Conditions Required for a RIC II to Drive This Signal           | RIC II Receiving the Signal                                                                  |
|----------------------|--------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------|----------------------------------------------------------------------------------------------|
| ACKI                 | Input signal to The PSM arbitration chain. This chain is employed to identify <i>PORT N</i> and <i>PORT M</i> . (Note 1) | Not applicable                                                  | This is dependent upon the method used to cascade RIC IIs, described in a following section. |
| ACKO                 | Output signal from the PSM arbitration chain.                                                                            | Not applicable                                                  | This is dependent upon the method used to cascade RIC IIs, described in a following section. |
| ACTN                 | This signal denotes there is activity on <i>PORT N</i> or <i>PORT M</i> .                                                | A RIC II must contain <i>PORT N</i> or <i>PORT M</i> . (Note 2) | The signal is monitored by all RIC IIs in the repeater system.                               |
| ANYXN                | This signal denotes that a repeater port that is not <i>PORT N</i> or <i>PORT M</i> is experiencing a collision.         | Any RIC II which satisfies the above condition. (Note 3)        | The signal is monitored by all RIC IIs in the repeater system.                               |
| COLN                 | Denotes <i>PORT N</i> or <i>PORT M</i> is experiencing a collision.                                                      | A RIC II must contain <i>PORT N</i> or <i>PORT M</i> . (Note 4) | The signal is monitored by all other RIC IIs in the repeater system.                         |
| IRE                  | This signal acts as an activity framing signal for the IRC and IRD signals.                                              | A RIC II must contain <i>PORT N</i> .                           | The signal is monitored by all other RIC IIs in the repeater system.                         |
| IRD                  | Decoded serial data, in NRZ format, received from the network segment attached to <i>PORT N</i> .                        | A RIC II must contain <i>PORT N</i> .                           | The signal is monitored by all other RIC IIs in the repeater system.                         |
| IRC                  | Clock signal associated with IRD and IRE.                                                                                | A RIC II must contain <i>PORT N</i> .                           | The signal is monitored by all other RIC IIs in the repeater system.                         |

**Note 1:** A RIC II which contains *PORT N* or *PORT M* may be identified by its  $\overline{\text{ACKO}}$  signal being low when its  $\overline{\text{ACKI}}$  input is high.

**Note 2:** Although this signal normally has only one source asserting the signal active it is used in a wired-or configuration.

**Note 3:** This bus line is used in a wired-or configuration.

**Note 4:** Refer to the note for the transmit collision case.

## 5.0 Functional Description (Continued)

### METHODS OF RIC II CASCADING

In order to build multi-RIC II repeaters, *PORT N* and *PORT M* identification must be performed across all the RIC IIs in the system. Inside each RIC II, the PSMs are arranged in a logical arbitration chain where port 1 is the highest and port 13 the lowest. The top of the chain, the input to port 1 is accessible to the user via the RIC II's  $\overline{\text{ACKI}}$  input pin. The output from the bottom of the chain becomes the  $\overline{\text{ACKO}}$  output pin. In a single RIC II system *PORT N* is defined as the highest port in the arbitration chain with receive or collision activity. *PORT N* identification is performed when the repeater is in the *IDLE* state. *PORT M* is defined as the highest port in the chain with a collision when the repeater leaves the *TRANSMIT COLLISION* state. In order for the arbitration chain to function, all that needs to be done is to tie the  $\overline{\text{ACKI}}$  signal to a logic high state. In multi-RIC II systems there are two methods to propagate the arbitration chain between RIC IIs:

The first and most straightforward is to extend the arbitration chain by daisy chaining the  $\overline{\text{ACKI}}$ - $\overline{\text{ACKO}}$  signals between RIC IIs. In this approach one RIC II is placed at the top of the chain (its  $\overline{\text{ACKI}}$  input is tied high), then the  $\overline{\text{ACKO}}$  signal from this RIC II is sent to the  $\overline{\text{ACKI}}$  input of the next RIC II and so on. This arrangement is simple to implement but it places some topological restrictions upon the repeater system. In particular, if the repeater is constructed using a backplane with removable printed circuit boards. (These boards contain the RIC IIs and their associated components.) If one of the boards is removed then the  $\overline{\text{ACKI}}$ - $\overline{\text{ACKO}}$  chain will be broken and the repeater will not operate correctly.

The second method of *PORT N* or *M* identification avoids this problem. This second technique relies on an external parallel arbiter which monitors all of the RIC IIs'  $\overline{\text{ACKO}}$  signals and responds to the RIC II with the highest priority. In this scheme each RIC II is assigned with a priority level. One method of doing this is to assign a priority number which reflects the position of a RIC II board on the repeater backplane, i.e., its slot number. When a RIC II experiences receive activity and the repeater system is in the *IDLE* state, the RIC II board will assert  $\overline{\text{ACKO}}$ . External arbitration logic drives the identification number onto an arbitration bus and the RIC II containing *PORT N* will be identified. An identical procedure is used in the *TRANSMIT COLLISION* state to identify *PORT M*. Parallel arbitration is not subject to the problems caused by missing boards, i.e., empty slots in the backplane. The logic associated with asserting this arbitration vector in the various packet repetition scenarios could be implemented in PAL® or GAL® type devices.

Both of the above methods employ the same signals:  $\overline{\text{ACKI}}$ ,  $\overline{\text{ACKO}}$  and  $\overline{\text{ACTN}}$  to perform *PORT N* or *M* arbitration.

The Inter-RIC bus allows multi-RIC II operations to be performed in exactly the same manner as if there is only a single RIC II in the system. The simplest way to describe the operation of Inter-RIC bus is to see how it is used in a number of common packet repetition scenarios. Throughout this description the RIC IIs are presumed to be operating in external transceiver mode. This is advantageous for the explanation since the receive, transmit and collision signals from each network segment are observable. In internal transceiver mode this is not the case, since the collision signal for the non-AUI ports is derived by the transceivers inside the RIC II.

### 5.4 EXAMPLES OF PACKET REPETITION SCENARIOS

The operation of RIC II is described by the following examples of packet repetition scenarios.

#### DATA REPETITION OVERVIEW

When a packet is received at one port, RIC II checks the source, and destination addresses of the packet. The port configuration causes either a pseudo random bit sequence, or the received packet to be transmitted to different ports.

If there is a destination address mismatch (secure mode), then the RIC II will generate a random pattern on the first bit of the data field on the corresponding transmitting port. The data remains intact on the Inter-RIC bus so other cascaded repeaters could compare the destination address with their local CAMs.

On a valid source address mismatch (secure mode), RIC II shall switch to random pattern both on the local transmitting ports and the Inter-RIC bus.

#### COLLISION SCENARIOS OVERVIEW

The RIC II will adhere to all collision scenarios. When a collision occurs, RIC II will switch from a random pattern to a jam pattern to comply with IEEE repeater specifications.

If collision occurs during the preamble, then jam pattern is transmitted out by the repeater's ports.

In the case of a collision during the address field, the "type field", or the "data field", RIC II switches to the jam pattern immediately.

#### FIFO CONDITION OVERVIEW

Elasticity buffer error (ELBER) or FIFO overflow burst is another condition that could take place anytime during the packet transmission. The sequence of events for FIFO burst is the same as those for collision.

#### DATA REPETITION PROCESS

The first task to be performed is *PORT N* identification. This is an arbitration process performed by the Port State Machines in the system. In situations where two or more ports simultaneously receive packets, the Inter-RIC bus operates by choosing one of the active ports, and forcing the others to transmit data (real data or pseudo random data). This is done in accordance with the IEEE specification's allowed exit paths from the *IDLE* state, i.e., to the *SEND PREAMBLE PATTERN* or *RECEIVE COLLISION* states.

The packet begins with a preamble pattern derived from the RIC II's on-chip jam/preamble generator. The data received at *PORT N* is directed through the receive multiplexor to the PLL decoder. Once phase lock has been achieved, the decoded data (in NRZ format) with its associated clock and enable signals, is asserted onto the IRD, IRE, and IRC Inter-RIC bus lines respectively. This serial data stream is received from the bus by all RIC IIs in the repeater and directed to their Elasticity Buffers. Logic circuits monitor the data stream and look for the Start of Frame Delimiter (SFD). When this has been detected, data is loaded into the elasticity buffer for later transmission. This will occur when sufficient preamble has been transmitted and certain internal state machine operations have been fulfilled.

Figure 5-3 shows two RIC IIs A and B, daisy chained together with RIC II A positioned at the top of the chain. A packet is received at port B1 of RIC II B, and is then repeated by the other ports in the system (non-secure mode). Figure 5-4



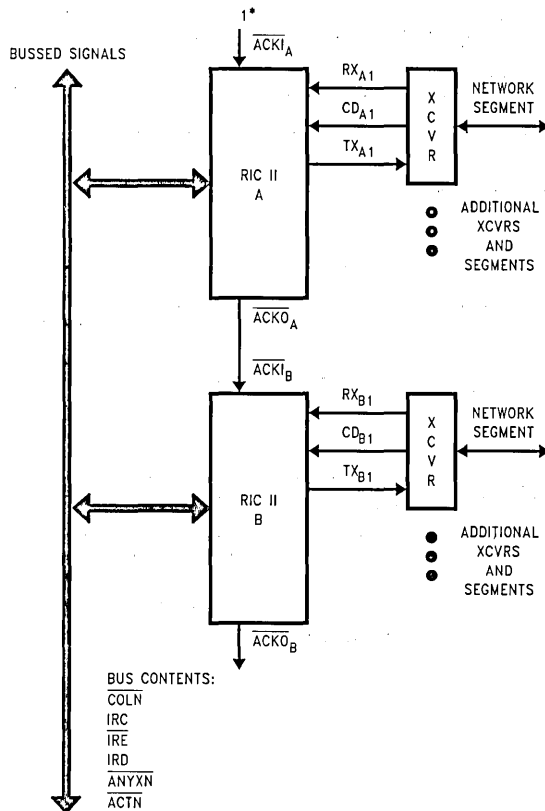
### 5.0 Functional Description (Continued)

shows the functional timing diagram for this packet repetition represented by the signals shown in *Figure 5-3*. In this example only two ports in the system are shown. In non-secure mode, the other ports also repeat the packet. It also indicates the operation of the RIC II's state machines in so far as can be seen by observing the Inter-RIC bus. For reference, the repeater's state transitions are shown in terms of the states defined by the IEEE specification. The location of *PORT N* is also shown. The following section describes the repeater and Inter-RIC bus transitions shown in *Figure 5-4*. The repeater is stimulated into activity by the data signal received by port B1. The RIC IIs in the system are alerted to forthcoming repeater operation by the falling edges on the  $\overline{ACKI}$  and  $\overline{ACKO}$  daisy chain and the  $\overline{ACTN}$  bus signal. Following a defined start up delay the repeater moves to the *SEND PREAMBLE* state. The RIC II system utilizes the start up delay to perform port arbitration. When packet transmis-

sion begins the RIC II system enters the *REPEAT* state. The expected, for normal packet repetition, sequence of repeater states, *SEND PREAMBLE*, *SEND SFD* and *SEND DATA* is followed, but is not visible upon the Inter-RIC bus. They are merged together into a single *REPEAT* state. This is also true for the *WAIT* and *IDLE* states, they appear as a combined Inter-RIC bus *IDLE* state.

Once a repeat operation has begun, i.e., the repeater leaves the *IDLE* state. It is required to transmit at least 96 bits of data or jam/preamble onto its network segments. If the duration of the received signal from *PORT N* is smaller than 96 bits, the repeater transitions to the *RECEIVE COLLISION* state (described later). This behavior is known as fragment extension.

After the packet data has been repeated, including the emptying of the RIC IIs' elasticity buffers, the RIC II performs the *Twt* transmit recovery operation. This is performed during the *WAIT* state shown in the repeater state diagram.

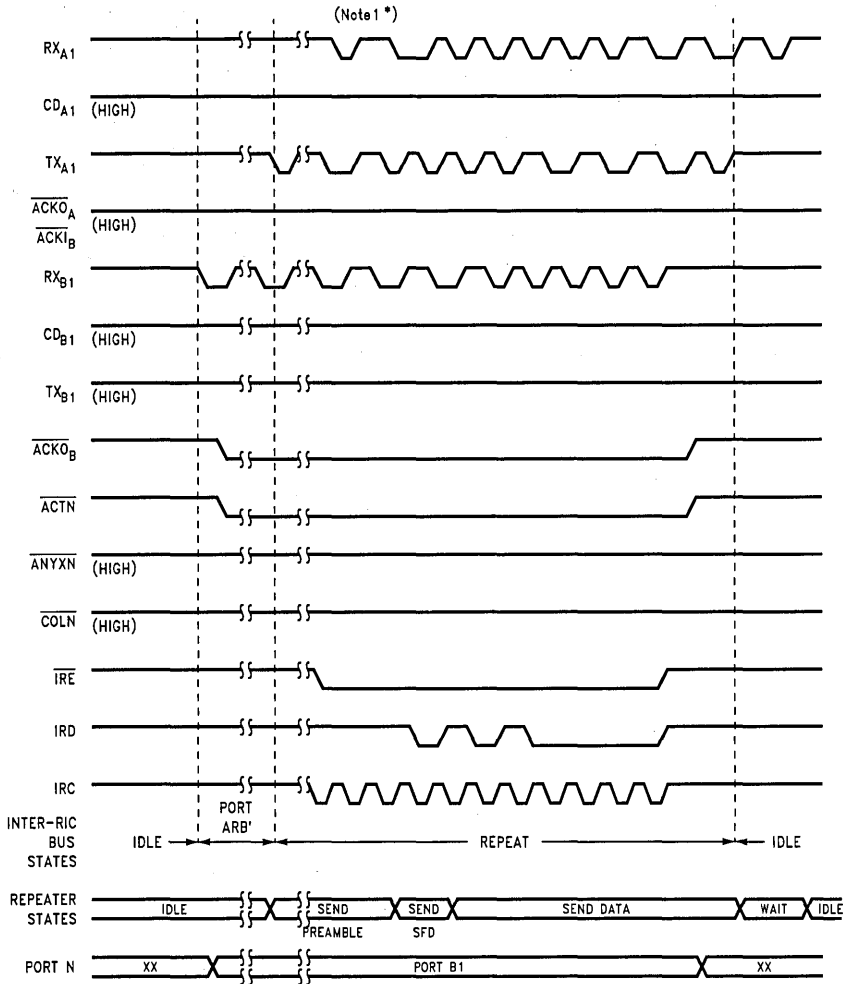


Note 1\*: This input is tied at a logic high state.

TL/F/12499-10

FIGURE 5-3. RIC II System Topology

## 5.0 Functional Description (Continued)

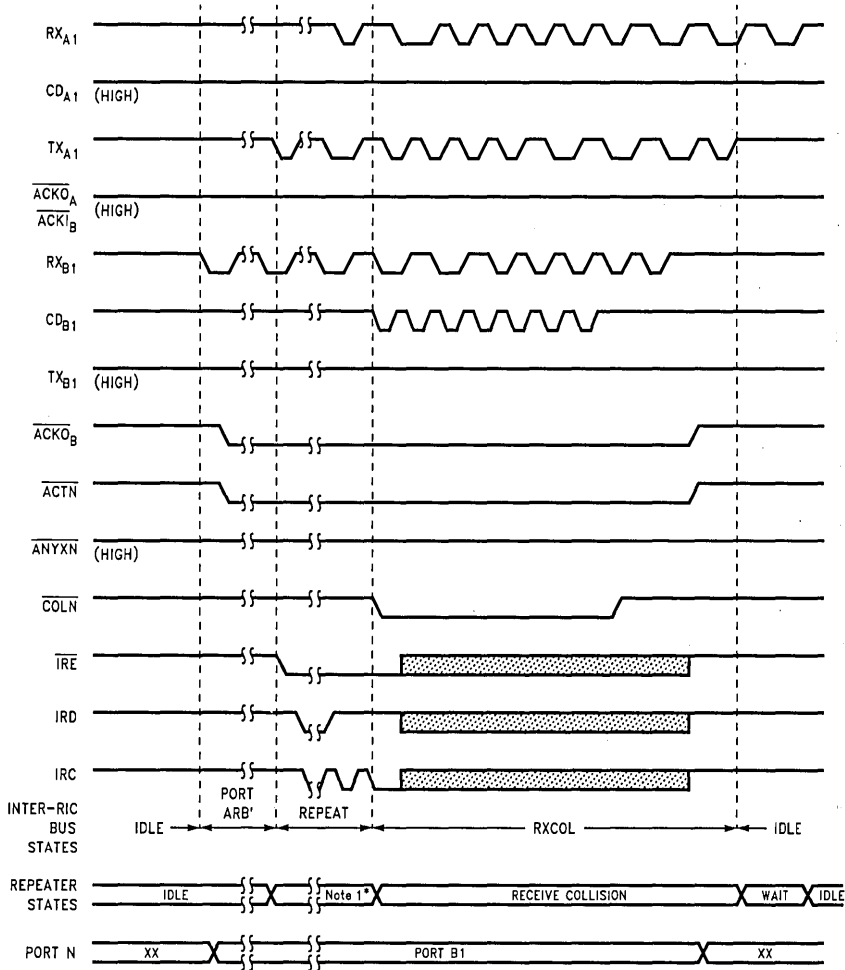


**Note 1\*:** The activity shown on RX<sub>A1</sub> represents the transmitted signal on TX<sub>A1</sub> after being looped back by the attached transceiver.

TL/F/12499-11

**FIGURE 5-4. Data Repetition**

### 5.0 Functional Description (Continued)



Note 1\*: SEND PREAMBLE, SEND SFD, SEND DATA

TL/F/12499-12

FIGURE 5-5. Receive Collision

## 5.0 Functional Description (Continued)

### RECEIVE COLLISIONS

A receive collision is a collision which occurs on the network segment attached to *PORT N*, i.e., the collision is "received" in a similar manner as a data packet is received, and then repeated to the other network segments. Not surprisingly, the receive collision propagation follows a similar sequence of operations as data repetition.

An arbitration process is performed to find *PORT N* and a preamble/jam pattern is transmitted by the repeater's other ports. When *PORT N* detects a collision on its segment the  $\overline{\text{COLN}}$  Inter-RIC bus signal is asserted. This forces all the RIC IIs in the system to transmit a preamble/jam pattern to their segments. This is important since they may be already transmitting data from their elasticity buffers. The repeater moves to the *RECEIVE COLLISION* state and begins to transmit the jam pattern. The repeater remains in this state until both the following conditions have been fulfilled:

1. at least 96 bits have been transmitted onto the network,
2. the activity has ended.

Under close examination, the repeater specification reveals that the actual end of activity has its own permutations of conditions:

1. collision and receive data signals may end simultaneously,
2. receive data may appear to end before collision signals,
3. receive data may continue for some time after the end of the collision signal.

Network segments using coaxial media may experience spurious gaps in segment activity when the collision signal goes inactive. This arises from the interaction between the receive and collision signal squelch circuits, implemented in coaxial transceivers, and the properties of the coaxial cable itself. The repeater specification avoids propagation of these activity gaps by extending collision activity by the  $T_{w2}$  wait time. Jam pattern transmission must be sustained throughout this period. After this, the repeater will move to the *WAIT* state unless there is a data signal being received by *PORT N*.

The functional timing diagram, *Figure 5-5*, shows the operation of a repeater system during a receive collision. The system configuration is the same as earlier described and is shown in *Figure 5-3*.

The RIC IIs perform the same *PORT N* arbitration and data repetition operations as previously described. The system is notified of the receive collision on port B1 by the  $\overline{\text{COLN}}$  bus signal going active. This is the signal which informs the main state machines to output the jam pattern rather than the data held in the elasticity buffers. Once a collision has occurred the IRC, IRD and  $\overline{\text{IRE}}$  bus signals may become undefined. When the collision has ended and the  $T_{w2}$  operation performed, the repeater moves to the *WAIT* state.

### TRANSMIT COLLISIONS

A transmit collision is a collision that is detected upon a segment to which the repeater system is transmitting. The port state machine monitoring the colliding segment asserts the  $\overline{\text{ANYXN}}$  bus signal. The assertion of  $\overline{\text{ANYXN}}$  causes *PORT M* arbitration to begin. The repeater moves to the

*TRANSMIT COLLISION* state when the port which had been *PORT N* starts to transmit a Manchester encoded 1 on to its network segment. While in the *TRANSMIT COLLISION* state, all ports of the repeater must transmit the 1010 . . . jam pattern, and *PORT M* arbitration is performed. Each RIC II is obliged, by the IEEE specification, to ensure all of its ports transmit for at least 96 bits once the *TRANSMIT COLLISION* state has been entered. This transmit activity is enforced by the  $\overline{\text{ANYXN}}$  bus signal. While  $\overline{\text{ANYXN}}$  is active, all RIC II ports will transmit jam. To ensure this situation lasts for at least 96 bits, the MSM inside the RIC IIs assert the  $\overline{\text{ANYXN}}$  signal throughout this period. After this period has elapsed,  $\overline{\text{ANYXN}}$  will only be asserted if there are multiple ports with active collisions on their network segments.

There are two possible ways for a repeater to leave the *TRANSMIT COLLISION* state. The most straightforward is when network activity, i.e., collisions and their  $T_{w2}$  extensions, end before the 96-bit enforced period expires. Under these conditions the repeater system may move directly to the *WAIT* state when 96 bits have been transmitted to all ports. If the MSM enforced period ends and there is still one port experiencing a collision, the *ONE PORT LEFT* state is entered. This may be seen on the Inter-RIC bus when  $\overline{\text{ANYXN}}$  is de-asserted and *PORT M* stops transmitting to its network segment. In this circumstance the Inter-RIC bus transitions to the *RECEIVE COLLISION* state. The repeater will remain in this state while *PORT M*'s collision,  $T_{w2}$  collision extension and any receive signals are present. When these conditions are not true, packet repetition finishes and the repeater enters the *WAIT* state.

*Figure 5-6* shows a multi-RIC II system operating under transmit collision conditions. There are many different scenarios which may occur during a transmit collision, this figure illustrates one of these. The diagram begins with packet reception by port A1. Port B1 experiences a collision, since it is not *PORT N* it asserts  $\overline{\text{ANYXN}}$ . This alerts the main state machines in the system to switch from data to jam pattern transmission.

Port A1 is also monitoring the  $\overline{\text{ANYXN}}$  bus line. Its assertion forces A1 to relinquish its *PORT N* status, start transmitting, stop asserting  $\overline{\text{ACTN}}$  and release its hold on the PSM arbitration signals ( $\overline{\text{ACKO}}$  A and  $\overline{\text{ACKI}}$  B). The first bit it transmits will be a Manchester encoded "1" in the jam pattern. Since port B1 is the only port with a collision, it attains *PORT M* status and stops asserting  $\overline{\text{ANYXN}}$ . It does however assert  $\overline{\text{ACTN}}$ , and exert its presence upon the PSM arbitration chain (forces  $\overline{\text{ACKO}}$  B low). The MSMs ensure that  $\overline{\text{ANYXN}}$  stays active and thus force all of the ports, including *PORT M*, to transmit to their segments.

After some time port A1 experiences a collision. This arises from the presence of the packet being received from port A1's segment and the jam signal the repeater is now transmitting onto this segment. Two packets on one segment results in a collision. *PORT M* now moves from B1 to A1. Port A1 fulfills the same criteria as B1, i.e., it has an active collision on its segment, but in addition it is higher in the arbitration chain. This priority yields no benefits for port A1 since the  $\overline{\text{ANYXN}}$  signal is still active. There are now two sources driving  $\overline{\text{ANYXN}}$ , the MSMs and the collision on port B1.

### 5.0 Functional Description (Continued)

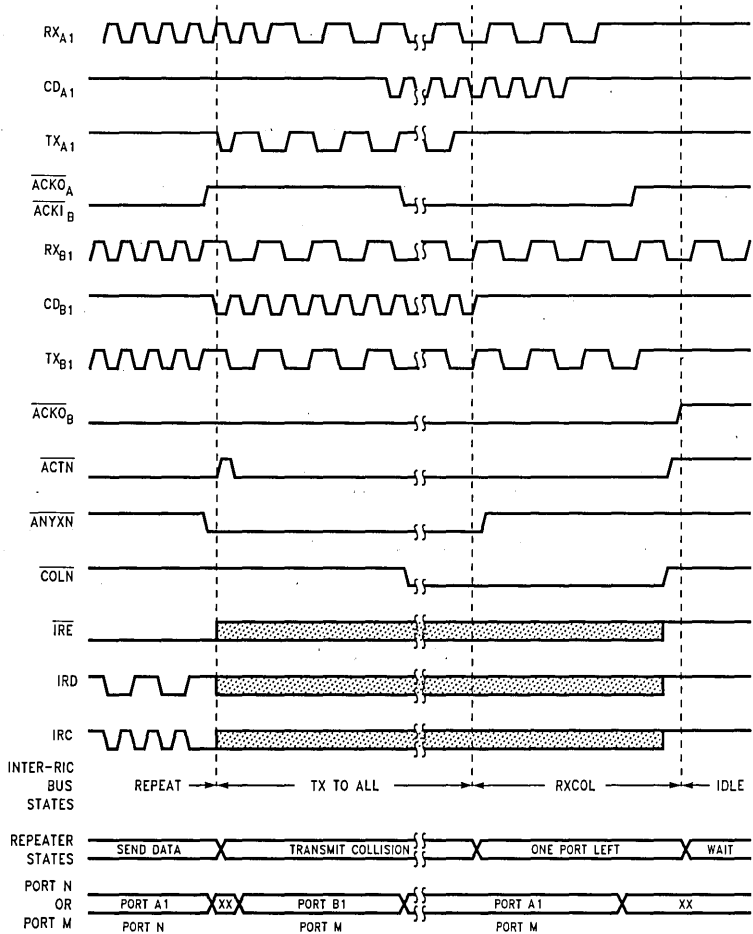
Eventually the collision on port B1 ends and the  $\overline{\text{ANYXN}}$  extension by the MSMs expires. There is only one collision on the network (this may be deduced since  $\overline{\text{ANYXN}}$  is inactive) so the repeater will move to the *ONE PORT LEFT* state. The RIC II system treats this state in a similar manner to a receive collision with *PORT M* fulfilling the role of the receiving port. The difference from a true receive collision is that the switch from packet data to the jam pattern has already been made (controlled by  $\overline{\text{ANYXN}}$ ). Thus the state of  $\overline{\text{COLN}}$  has no effect upon repeater operations. In common with the operation of the *RECEIVE COLLISION* state, the repeater remains in this condition until the collision and receive activity on *PORT M* subside. The packet repetition operation completes when the *Tw1* recovery time in the *WAIT* state has been performed.

**Note:** In transmit collision conditions  $\overline{\text{COLN}}$  will only go active if the RIC II which contained *PORT N* at the start of packet repetition contains *PORT M* during the *TRANSMIT COLLISION* and *ONE PORT LEFT* states.

### JABBER PROTECTION

A repeater is required to disable transmit activity if the length of its current transmission reaches the jabber protect limit. This is defined by the specification's *Tw3* time. The repeater disables output for a time period defined by the *Tw4* specification, after this period normal operation may resume.

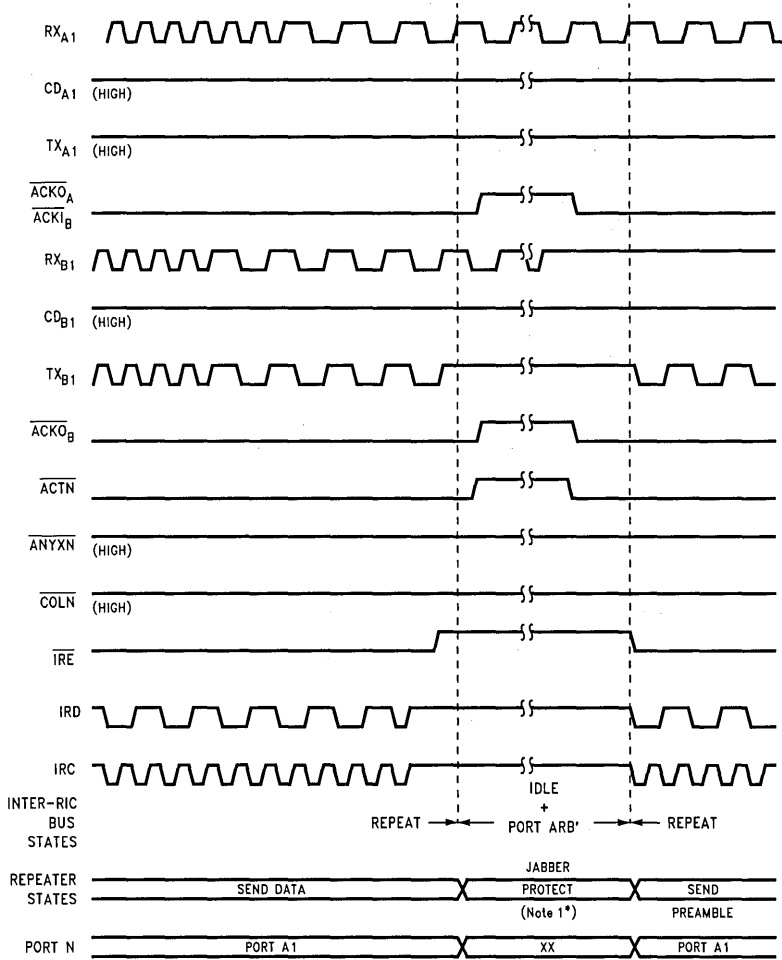
Figure 5-7 shows the effect of a jabber length packet upon a RIC II based repeater system. The **JABBER PROTECT** state is entered from the *SEND DATA* state. While the *Tw4* period is observed the Inter-RIC bus displays the *IDLE* state. This is misleading since new packet activity or continuous activity (as shown in the diagram) does not result in packet repetition. This may only occur when the *Tw4* requirement has been satisfied.



**Note:** The Inter-RIC bus is configured to use active low signals.

FIGURE 5-6. Transmit Collision

### 5.0 Functional Description (Continued)



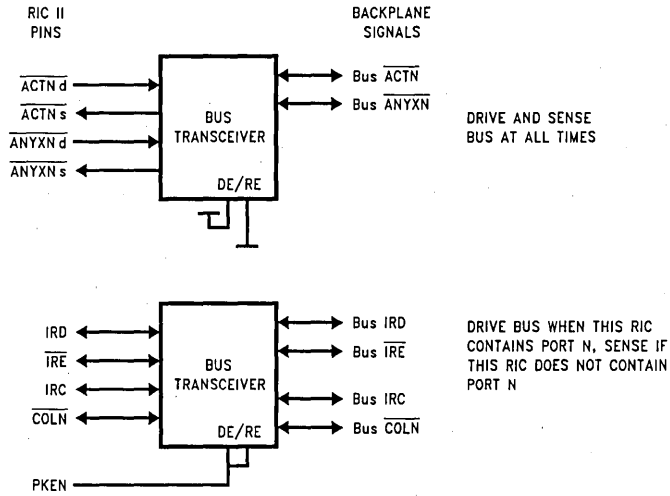
TL/F/12499-14

**Note 1\*:** The IEEE Specification does not have a jabber protect state defined in its main state diagram, this behavior is defined in an additional MAU Jabber Lockup Protection state diagram.

**Note:** The Inter-RIC bus is configured to use active low signals.

**FIGURE 5-7. Jabber Protect**

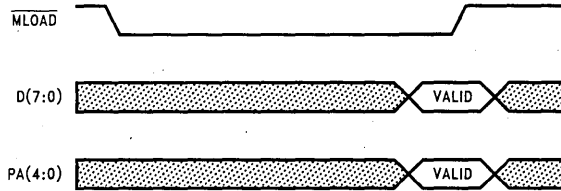
### 5.0 Functional Description (Continued)



TL/F/12499-15

**Note:** DE = Bus Drive Enable active high,  $\overline{RE}$  = Bus Receive Enable active low  
**Note:** The Inter-RIC bus is configured to use active low signals.

**FIGURE 5-8. External Bus Transceiver Connection Diagram**



TL/F/12499-16

**FIGURE 5-9. Mode Load Operation**

## 5.0 Functional Description (Continued)

### 5.5 DESCRIPTION OF HARDWARE CONNECTION FOR INTER-RIC BUS

When considering the hardware interface, the Inter-RIC bus may be viewed as consisting of three groups of signals:

1. Port Arbitration chain, namely:  $\overline{ACKI}$  and  $\overline{ACKO}$ .
2. Simultaneous drive and sense signals, i.e.,  $\overline{ACTN}$  and  $\overline{ANYXN}$ . (Potentially these signals may be driven by multiple devices.)
3. Drive or sense signals, i.e.,  $\overline{IRE}$ , IRD, IRC and  $\overline{COLN}$ . (Only one device asserts these signals at any instance in time.)

The first set of signals are either used as point to point links, or with external arbitration logic. In both cases the load on these signals will not be large, so the on-chip drivers are adequate. This may not be true for signal classes (2) and (3).

The Inter-RIC bus has been designed to connect RIC II's together directly, or via external bus transceivers. The latter is advantageous in large repeaters. In the second application the backplane is often heavily loaded and is beyond the drive capabilities of the on-chip bus drivers. The need for simultaneous sense and drive capabilities on the  $\overline{ACTN}$  and  $\overline{ANYXN}$  signals, and the desire to allow operation with external bus transceivers, makes it necessary for these bus signals to each have a pair of pins on the RIC II. One driving the bus, the other sensing the bus signal. When external bus transceivers are used, they must be open collector/open drain to allow wire-ORing of the signals. Additionally, the drive and sense enables of the bus transceiver should be tied in the active state.

The uni-directional nature of information transfer on the  $\overline{IRE}$ , IRD, IRC and  $\overline{COLN}$  signals, means a RIC II is either driving these signals or receiving them from the bus, but not both at the same time. Thus a single bi-directional input/output pin is adequate for each of these signals. If an external bus transceiver is used with these signals the Packet Enable "PKEN" RIC II output pin performs the function of a drive enable and sense disable.

*Figure 5-8* shows the RIC II connected to the Inter-RIC bus via external bus transceivers, such as National's DS3893A bus transceivers.

Some bus transceivers are of the inverting type. To allow the Inter-RIC bus to utilize these transceivers, the RIC II may be configured to invert the active states of the  $\overline{ACTN}$ ,  $\overline{ANYXN}$ ,  $\overline{COLN}$  and  $\overline{IRE}$  signals. Instead of being active low they are active high. Thus they become active low once more when passed through an inverting bus driver. This is particularly important for the  $\overline{ACTN}$  and  $\overline{ANYXN}$  bus lines, since these signals must be used in a wired-or configuration. Incorrect signal polarity would make the bus unusable.

### 5.6 PROCESSOR AND DISPLAY INTERFACE

The processor interface pins, which include the data bus, address bus and control signals, actually perform three operations which are multiplexed on these pins. These operations are:

1. The Mode Load Operation, which performs a power up initialization cycle upon the RIC II.
2. Display Update Cycles, which are refresh operations for updating the display LEDs.

3. Processor Access Cycles, which allows  $\mu P$ 's to communicate with the RIC II's registers.

These three operations are described below.

#### MODE LOAD OPERATION

The Mode Load Operation is a hardware initialization procedure performed at power on. It loads vital device configuration information into on-chip configuration registers. In addition to its configuration function, the  $\overline{MLOAD}$  pin is the RIC II's reset input. When  $\overline{MLOAD}$  is low all of the RIC II's repeater timers, state machines, segment partition logic and hub management logic are reset.

The Mode Load Operation may be accomplished by attaching the appropriate set of pull up and pull down resistors to the data and register address pins to assert logic high or low signals onto these pins, and then providing a rising edge on the  $\overline{MLOAD}$  pin as is shown in *Figure 5-9*. The mapping of chip functions to the configuration inputs is shown in Table 5-1. Such an arrangement may be performed using a simple resistor, capacitor, diode network. Performing the Mode Load Operation in this way enables the configuration of a RIC II that is in a simple repeater system (one without a processor).

Alternatively in a complex repeater system, the Mode Load Operation may be performed using a processor write cycle. This would require the  $\overline{MLOAD}$  pin be connected to the CPU's write strobe via some decoding logic, and included in the processor's memory map.

To support the security options, pin D0 of  $\overline{MLOAD}$  is assigned to configure RIC II during mode load operation. A pull up (non-security mode) or a pull down (security mode) on this pin defines the desired security level. By using this bit, the user could also take advantage of the learning mode, as described below.

#### LEARNING OF PORT SOURCE ADDRESS(ES)

Learning mode could be invoked in two ways according to bit D0 of  $\overline{MLOAD}$  configuration. Only the port CAMs are capable of learning the addresses:

1. When D0=0, upon power up and by default, LME, SME, ESA and EDA bits in the Port Security Configuration Register (PSCR) are set globally. This means that each port will learn the address of the node connected to it by the reception of the first good packet. The second address is learned only if it is different than the first one. Only the address of a packet with correct CRC can be learned. As soon as the address is learned by any of the two CAM locations, RIC II will set the corresponding ADV (Address Valid) bit in Port CAM Pointer Register.

To start the address comparison, the SAC (Start Comparison) bit must be set (SAC=1) by the user. RIC II will only use this CAM location for comparison when the ADV bit is set (ADV=1), whether LME is 1 or 0. These four bits in PSCR could be disabled later on a per port basis, which allow all the packets regardless of their address to pass through the repeater.

2. When D0=1 for  $\overline{MLOAD}$ , security could still be done, but this time it means that the user should set the LME, SME, ESA and/or EDA bits in the Port Security Configuration Register. The rest of the operation is same as when D0 is equal to zero.



## 5.0 Functional Description (Continued)

It is important to note that RIC II will learn the address of the packet if LME is set regardless of the D0 setting of MLOAD, i.e. secure or non-secure mode.

It is also very important to note that for proper address learning, LME and SAC should not be set together.

When the repeater is in non-secure mode, then the comparison will not take place between the incoming address and the learned address.

When the repeater is in secure mode, and the LME bit is set, then the processor read/write access will be ignored for the port CAM entries. That is read/write cycles are completed, however unknown values are read during the learning process. Data will not be written into the CAM entries until the end of the learning process.

It may be desired not to randomize the outgoing data and transmit the data intact when there is a valid source address mismatch. The Generate Random Pattern bit, GRP in the Global Security Register, will provide the option.

If GRP is set (GRP = 1) and there is a source address mismatch, then RIC II will not generate random pattern; the packet will be transmitted out and the Hub Manager will be informed about the source address mismatch.

For this option to work properly, GRP = ESA = 1 and EDA = 0. If EDA is also set to 1, then the packet will be randomized on ports with valid DA mismatches, and this functionality will not work.

**TABLE 5-1. Pin Definitions for Options in the Mode Load Operation**

| Pin Name | Programming Function | Effect When Bit is 0 | Effect When Bit is 1 | Function                                                                                                                                                                                                                                                                                                                                                         |
|----------|----------------------|----------------------|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0       | SCRTY                | Security Mode        | Non-Security Mode    | This bit configures RIC II security feature options. When D0 = 0 LME, SME, ESA, EDA bits in the Port Security Configuration Register (PSCR) are set globally. When D0 = 1 security can still be done, but now the user needs to set the above bits in the PSCR register.                                                                                         |
| D1       | TW2                  | 5 Bits               | 3 Bits               | This allows the user to select one of two values for the repeater specification TW2 time. The lower limit (3 bits) meets the IEEE specification. The upper limit (5 bits) is not specification compliant but may provide users with higher network throughput by avoiding spurious network activity gaps when using coaxial (10BASE2, 10BASE5) network segments. |
| D2       | CLIM                 | 63                   | 31                   | The partition specification requires a port to be partitioned after a certain number of consecutive collisions. The RIC II has two values available to allow users to customize the partitioning algorithm to their environment. Please refer to the Partition State Machine, in data sheet Section 7.3.                                                         |
| D3       | LPPART               | Selected             | Not Selected         | The RIC II may be configured to partition a port if the segment transceiver does not loopback data to the port when the port is transmitting to it, as described in the Partition State Machine.                                                                                                                                                                 |
| D4       | OWCE                 | Selected             | Not Selected         | This configuration bit allows the on-chip partition algorithm to include out of window collisions into the collisions it monitors, as described in the Partition State Machine.                                                                                                                                                                                  |
| D5       | TXONLY               | Selected             | Not Selected         | This configuration bit allows the on-chip partition algorithm to restrict segment reconnection, as described in the Partition State Machine.                                                                                                                                                                                                                     |
| D6       | DPART                | Selected             | Not Selected         | The Partition state machines for all ports may be disabled by writing a logic zero to this bit during the mode load operation.                                                                                                                                                                                                                                   |
| D7       | MIN/MAX              | Minimum Mode         | Maximum Mode         | The operation of the display update block is controlled by the value of this configuration bit, as described in the Display Update Cycles section.                                                                                                                                                                                                               |

## 5.0 Functional Description (Continued)

TABLE 5-1. Pin Definitions for Options in the Mode Load Operation (Continued)

| Pin Name   | Programming Function | Effect When Bit Is 0                                                                     | Effect When Bit Is 1 | Function                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
|------------|----------------------|------------------------------------------------------------------------------------------|----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|--------|-------------|---|---|-------------------------------------------------------------|---|---|------------------------------------------------------------------------------------------|---|---|------------------------------------------------------------------------------------------|---|---|-------------------------------------------------------------|
| RA0<br>RA1 | BYPAS1<br>BYPAS2     |                                                                                          |                      | <p>These configuration bits select which of the repeater ports (numbers 2 to 13) are configured to use the on-chip internal 10BASE-T transceivers or the external transceiver interface. The external transceiver interface operates using AUI compatible signal levels.</p> <table border="1"> <thead> <tr> <th>BYPAS2</th> <th>BYPAS1</th> <th>Information</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>All ports (2 to 13) use the external Transceiver Interface.</td> </tr> <tr> <td>0</td> <td>1</td> <td>Ports 2 to 5 use the external interface, 6 to 13 use the internal 10BASE-T transceivers.</td> </tr> <tr> <td>1</td> <td>0</td> <td>Ports 2 to 7 use the external interface, 8 to 13 use the internal 10BASE-T transceivers.</td> </tr> <tr> <td>1</td> <td>1</td> <td>All ports (2 to 13) use the internal 10BASE-T transceivers.</td> </tr> </tbody> </table> | BYPAS2 | BYPAS1 | Information | 0 | 0 | All ports (2 to 13) use the external Transceiver Interface. | 0 | 1 | Ports 2 to 5 use the external interface, 6 to 13 use the internal 10BASE-T transceivers. | 1 | 0 | Ports 2 to 7 use the external interface, 8 to 13 use the internal 10BASE-T transceivers. | 1 | 1 | All ports (2 to 13) use the internal 10BASE-T transceivers. |
| BYPAS2     | BYPAS1               | Information                                                                              |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
| 0          | 0                    | All ports (2 to 13) use the external Transceiver Interface.                              |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
| 0          | 1                    | Ports 2 to 5 use the external interface, 6 to 13 use the internal 10BASE-T transceivers. |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
| 1          | 0                    | Ports 2 to 7 use the external interface, 8 to 13 use the internal 10BASE-T transceivers. |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
| 1          | 1                    | All ports (2 to 13) use the internal 10BASE-T transceivers.                              |                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
| RA2        | BINV                 | Active High Signals                                                                      | Active Low Signals   | This selection determines whether the Inter-RIC signals: IRE, ACTN, ANYXN, COLN and Management bus signal MCRS are active high or low.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
| RA3        | EXPLL                | External PLL                                                                             | Internal PLL         | If desired, the RIC II may be used with an external decoder, this configuration bit performs the selection.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |
| RA4        | resv                 | Not Permitted                                                                            | Required             | To ensure correct device operation, this bit must be written with a logic one during the mode load operation.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |        |        |             |   |   |                                                             |   |   |                                                                                          |   |   |                                                                                          |   |   |                                                             |

## 5.0 Functional Description (Continued)

### 5.7 DESCRIPTION OF HARDWARE CONNECTION FOR PROCESSOR AND DISPLAY INTERFACE

#### DISPLAY UPDATE CYCLES

The RIC II possesses control logic and interface pins which may be used to provide status information concerning activity on the attached network segments and the current status of repeater functions. These status cycles are completely autonomous and require only simple support circuitry to produce the data in a form suitable for a light emitting diode "LED" display. The display may be used in one of two modes:

1. Minimum mode—General Repeater Status LEDs,
2. Maximum mode—Individual Port Status LEDs.

Minimum mode, intended for simple LED displays, makes available four status indicators. The first LED denotes whether the RIC II has been forced to activate its jabber protect functions. The remaining 3 LEDs indicate if any of the RIC II's network segments are: (1) experiencing a collision, (2) receiving data, (3) currently partitioned. When minimum display mode is selected the only external components required are a 74LS374 type latch, the LEDs and their current limiting resistors.

Maximum mode differs from minimum mode by providing display information specific to individual network segments. This information denotes the collision activity, packet reception and partition status of each segment. In the case of 10BASE-T segments the link integrity status and polarity of the received data are also made available. The wide variety of information available in maximum mode may be used in its entirety or in part. Thus allowing the system designer to choose the appropriate complexity of status display commensurate with the specification of the end equipment.

The signals provided and their timing relationships have been designed to interface directly with 74LS259 type addressable latches. The number of latches used being dependent upon the complexity of the display. Since the latches are octal, a pair of latches is needed to display each type of segment specific data (13 ports means 13 latch bits). The accompanying tables (5-2 and 5-3) show the function of the interface pins in minimum and maximum modes. *Figure 5-11* shows the location of each port's status information when maximum mode is selected. This may be compared with the connection diagram *Figure 5-10*.

Immediately following the Mode Load Operation (when the MLOAD pin transitions to a high logic state), the display logic performs an LED test operation. This operation lasts one second. While it is in effect, all of the utilized LEDs will blink on. Thus, an installation engineer is able to test the operation of the display by forcing the RIC II into a reset cycle (MLOAD forced low). The rising edge on the MLOAD pin starts the LED test cycle. **During the LED test cycle the RIC II does not perform packet repetition operations.**

The status display possesses a capability to lengthen the time an LED is active. At the end of the repetition of a packet, the display is frozen showing the current activity. This freezing lasts for 30 ms or until a subsequent packet is repeated. Thus at low levels of packet activity, the display stretches activity information to make it discernible to the human eye. At high traffic rates the relative brightness of the LEDs indicates those segments with high or low activity.

It should be mentioned that when the Real Time Interrupt (RTI) occurs, the display update cycle will stop and after RTI is serviced, the display update cycle will resume activity.

## 5.0 Functional Description (Continued)

**TABLE 5-2. Status Display Pin Functions in Minimum Mode**

| Signal Pin Name | Function in Minimum Mode                                                                                                                             |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0              | No operation                                                                                                                                         |
| D1              | Provides status information indicating if there is a collision occurring on one of the segments attached to this RIC II.                             |
| D2              | Provides status information indicating if one of this RIC II's ports is receiving a data or collision packet from a segment attached to this RIC II. |
| D3              | Provides status information indicating that the RIC II has experienced a jabber protect condition.                                                   |
| D4              | Provides Status information indicating if one of the RIC II's segments is partitioned.                                                               |
| D(7:5)          | No operation.                                                                                                                                        |
| STR0            | This signal is the latch enable for the 374 type latch.                                                                                              |
| STR1            | This signal is held at a logic one.                                                                                                                  |

**TABLE 5-3. Status Display Pin Functions in Maximum Mode**

| Signal Pin Name | Function in Maximum Mode                                                                                                                                                                                                                                                                                  |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0              | Provides status information concerning the Link Integrity status of 10BASE-T segments. This signal should be connected to the data inputs of the chosen pair of 74LS259 latches.                                                                                                                          |
| D1              | Provides status information indicating if there is a collision occurring on one of the segments attached to this RIC II. This signal should be connected to the data inputs of the chosen pair of 74LS259 latches.                                                                                        |
| D2              | Provides status information indicating if one of this RIC II's ports is receiving a data or a collision packet from its segment. This signal should be connected to the data inputs of the chosen pair of 74LS259 latches.                                                                                |
| D3              | Provides Status information indicating that the RIC II has experienced a jabber protect condition. Additionally it denotes which of its ports are partitioned. This signal should be connected to the data inputs of the chosen pair of 74LS259 latches.                                                  |
| D4              | Provides status information indicating if one of this RIC II's ports is receiving data of inverse polarity. This status output is only valid if the port is configured to use its internal 10BASE-T transceiver. The signal should be connected to the data inputs of the chosen pair of 74LS259 latches. |
| D(7:5)          | These signals provide the repeater port address corresponding to the data available on D(4:0).                                                                                                                                                                                                            |
| STR0            | This signal is the latch enable for the lower byte latches, that is the 74LS259s which display information concerning ports 1 to 7.                                                                                                                                                                       |
| STR1            | This signal is the latch enable for the upper byte latches, that is the 74LS259s which display information concerning ports 8 to 13.                                                                                                                                                                      |

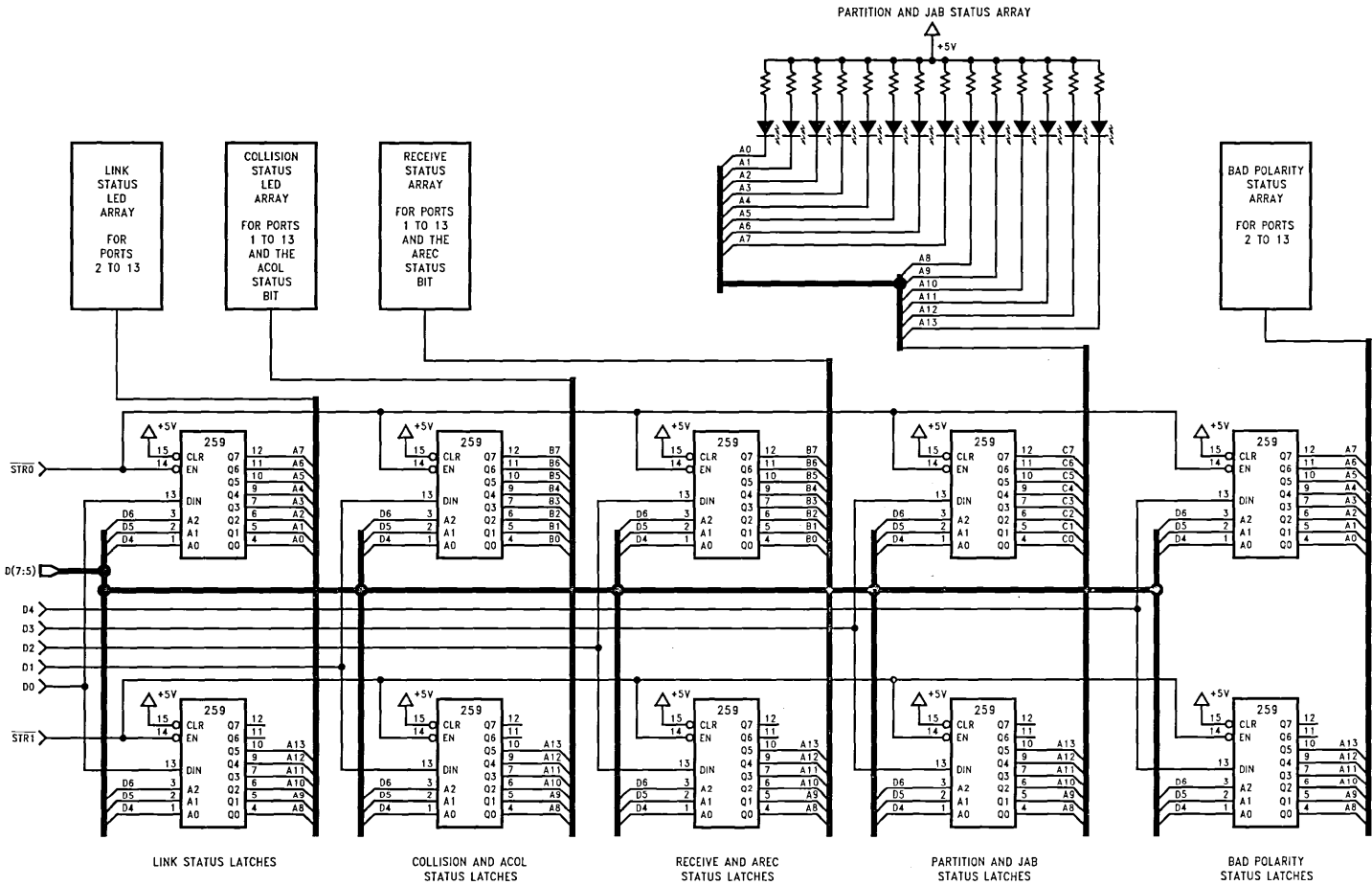


FIGURE 5-10. Maximum Mode LED Display (All Available Status Bits Used)

TL/F/12499-17

## 5.0 Functional Description (Continued)

74LS259 Latch Inputs =  $\overline{STR0}$

| 259 Output         | Q0   | Q1      | Q2    | Q3    | Q4    | Q5    | Q6    | Q7    |
|--------------------|------|---------|-------|-------|-------|-------|-------|-------|
| 259 Addr S2-0      | 000  | 001     | 010   | 011   | 100   | 101   | 110   | 111   |
| RIC II Port Number |      | 1 (AUI) | 2     | 3     | 4     | 5     | 6     | 7     |
| RIC II D0 259 # 1  |      |         | LINK  | LINK  | LINK  | LINK  | LINK  | LINK  |
| RIC II D1 259 # 2  | ACOL | COL     | COL   | COL   | COL   | COL   | COL   | COL   |
| RIC II D2 259 # 3  | AREC | REC     | REC   | REC   | REC   | REC   | REC   | REC   |
| RIC II D3 259 # 4  | JAB  | PART    | PART  | PART  | PART  | PART  | PART  | PART  |
| RIC II D4 259 # 5  |      |         | BDPOL | BDPOL | BDPOL | BDPOL | BDPOL | BDPOL |

74LS259 (or Equiv.) Latch Inputs =  $\overline{STR1}$

| 259 Outputs        | Q0    | Q1    | Q2    | Q3    | Q4    | Q5    | Q6  | Q7  |
|--------------------|-------|-------|-------|-------|-------|-------|-----|-----|
| 259 Addr S2-0      | 000   | 001   | 010   | 011   | 100   | 101   | 110 | 111 |
| RIC II Port Number | 8     | 9     | 10    | 11    | 12    | 13    |     |     |
| RIC II D0 259 # 6  | LINK  | LINK  | LINK  | LINK  | LINK  | LINK  |     |     |
| RIC II D1 259 # 7  | COL   | COL   | COL   | COL   | COL   | COL   |     |     |
| RIC II D2 259 # 8  | REC   | REC   | REC   | REC   | REC   | REC   |     |     |
| RIC II D3 259 # 9  | PART  | PART  | PART  | PART  | PART  | PART  |     |     |
| RIC II D4 259 # 10 | BDPOL | BDPOL | BDPOL | BDPOL | BDPOL | BDPOL |     |     |

This shows the LED Output Functions for the LED Drivers when 74LS259s are used. The top table refers to the bank of 4 74LS259s latched with  $\overline{STR0}$ , and the lower table refers to the bank of 4 74LS259s latched with  $\overline{STR1}$ . (For example the RIC II's D0 data signal goes to 259 #1 and #5. These two 74LS259s then drive the LINK LEDs.)

**Note:** ACOL = Any Port Collision, AREC = Any Port Reception, JAB = Any Port Jabbering, LINK = Port Link, COL = Port Collision, REC = Port Reception, PART = Port Partitioned, BDPOL = Bad (inverse) Polarity of received data.

**FIGURE 5-11. Maximum Mode LED Definitions**

5.0 Functional Description (Continued)

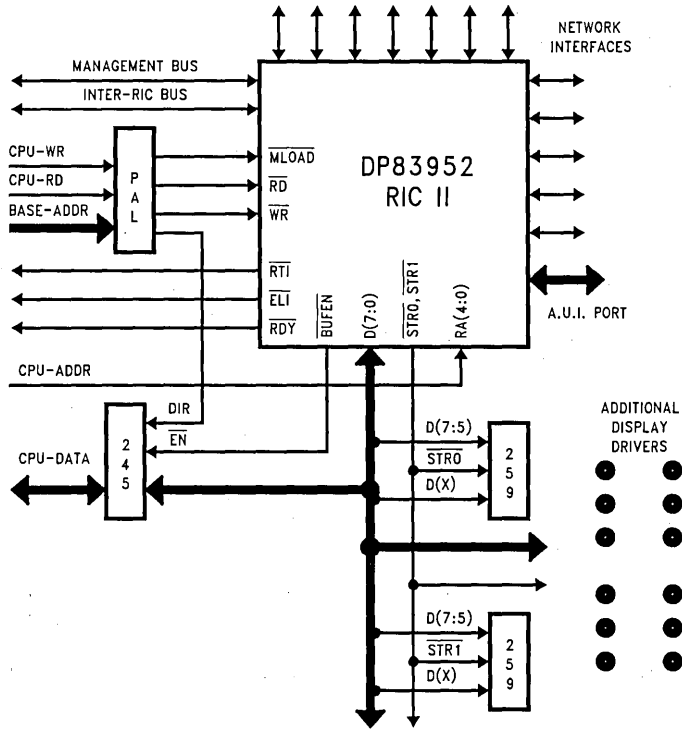


FIGURE 5-12. Processor Connection Diagram

TL/F/12499-18

## 5.0 Functional Description (Continued)

### PROCESSOR ACCESS CYCLES

Access to the RIC II's on-chip registers is made via its processor interface. This utilizes conventional non-multiplexed address (5-bit) and data (8-bit) busses. The data bus is also used to provide data and address information to off-chip display latches during display update cycles. While performing these cycles, the RIC II behaves as a master of its data bus. Consequently a TRI-STATE bi-directional bus transceiver, e.g., 74LS245 must be placed between the RIC II and any processor bus.

RIC II provides the following scheme to facilitate faster register accesses. Lower Event Information Register (Page 1H, Address 1FH) has Disable LED Update bit, DLU, which when it is set, then RIC II stops the LED updates. This causes the data bus to be no longer shared, and therefore RIC II is always in a slave access mode. In this mode, the maximum read/write cycle time is reduced to approximately 400 ns.

The processor requests a register access by asserting the read "RD" or write "WR" input strobes. The RIC II responds by finishing any current display update cycle and asserts the TRI-STATE buffer enable signal "BUFEN". If the processor cycle is a write cycle then the RIC II's data buffers are disabled to prevent contention. In order to interface to the RIC II in a processor controlled system it is likely a PAL device will be used to perform the following operations:

1. locate the RIC II in the processor's memory map (address decode),
2. generate the RIC II's read and write strobes,
3. control the direction signal for the 74LS245.

An example of the processor and display interfaces is shown in *Figure 5-12*.

### INTERRUPT HANDLING

The DP83952 RIC II offers an alternate method for a faster access to determine the source of the Event Logging Interrupt (ELI) then the DP83950 RIC.

For an event logging interrupt due to flag found, the **DP83950 RIC** requires the following scheme:

1. Read the Page Select Register (Address 10H) to locate the source of Event Logging Interrupt.
2. Read all the Port Event Recording Registers (Page 1H, Address 11H to 1DH) to find the port and the event responsible for Event Logging Interrupt.

**DP83952 RIC II** allows the following alternate scheme for a faster access:

1. Read Page Select Register (Address 10H) to locate the source of Event Logging Interrupt.
2. Read the Event Information Registers (Page 1H, addresses 1EH and 1FH) to locate the port responsible for interrupt.
3. Read the Event Recording register of that port to find which specific event caused the Event Logging Interrupt.

## 6.0 Hub Management Support

The RIC II provides information regarding the status of its ports and the packets it is repeating. This data is available in three forms:

1. Counted Events—Network events accumulated into the RIC II's 16-bit Event Counter Registers.
2. Recorded Events—Network events that set bits in the Event Record Registers.
3. Hub Management Status packets—This is information sent over the Management Bus in a serial function to be decoded by an Ethernet Controller board.

The counted and recorded event information is available through the processor interface. This data is port specific and may be used to generate interrupts via the Event Logging Interrupt "ELI" pin. Since the information is specific to each port, each repeater port has its own event record register and event counter. The counters and event record registers have user definable masks which enable them to be configured to count and record a variety of events. The counters and record registers are designed to be used together, so that detailed information, i.e., a count value can be held on-chip for a specific network condition. More general information, i.e. the occurrence of certain types of events may be retained in on-chip latches. Thus, the user can configure the counters to increment upon a rapidly occurring event (most likely to be used to count collisions), and the record registers may log the occurrence of less frequent error conditions such as jabber protect packets.

### 6.1 EVENT COUNTING FUNCTION

The counters may increment upon the occurrence of one of the categories of event as described below.

Potential sources for Counter increment:

**Jabber Protection (JAB):** The port counter increments if the length of a received packet from its associated port, causes the repeater state machine to enter the jabber protect state.

**Elasticity Buffer Error (ELBER):** The port counter increments if a Elasticity Buffer underflow or overflow occurs during packet reception. The flag is held inactive if a collision occurs during packet reception or if a phase lock error, described below, has already occurred during the repetition of the packet.

**Phase Lock Error (PLER):** A phase lock error is caused if the phase lock loop decoder loses lock during packet reception. Phase lock onto the received data stream may or may not be recovered later in the packet, and data errors may have occurred. This flag is held inactive if a collision occurs.

**Non-SFD Packet (NSFD):** If a packet is received, and the start of frame delimiter is not found, the port counter will increment. Counting is inhibited if the packet suffers a collision.

**Out of Window Collision (OWC):** The out of window collision flag for a port goes active when a collision is experienced outside of the network slot time.

**Transmit Collision (TXCOL):** The transmit collision flag for a port is enabled when a transmit collision is experienced by the repeater. Each port experiencing a collision under these conditions is said to have suffered a transmit collision.



## 6.0 Hub Management Support (Continued)

**Receive Collision (RXCOL):** The receive collision flag for a port goes active when the port is the receive source of network activity and suffers a collision, provided no other network segments experience collisions then the receive collision flag for the receiving port will be set.

**Partition (PART):** The port counter increments when a port becomes partitioned.

**Bad Link (BDLNK):** The port counter increments when a port is configured for 10BASE-T operation has entered the link lost state.

**Short Event reception (SE):** The port counter increments if the received packet is less than 74 bits long and no collision occurs during reception.

**Packet Reception (REC):** When a packet is received the port counter increments.

In order to utilize the counters the user must choose, from the above list, the desired statistic for counting. This counter mask information must be written to the appropriate, Event Count Mask Register. There are two of these registers, the Upper and Lower Event Count Mask registers. For the exact bit patterns of these registers please see Section 8 of the data sheet.

For example if the counters are configured to count network collisions and the appropriate masks have been set, then whenever a collision occurs on a segment, this information is latched by the hub management support logic. At the end of repetition of the packet the collision status, respective to each port, is loaded into that port's counter. This operation is completely autonomous and requires no processor intervention.

Each counter is 16 bits long and may be directly read by the processor. Additionally each counter has a number of decodes to indicate the current value of the count. There are three decodes:

low count (a value of 00FF Hex and under),  
high count (a value of C000 Hex and above),  
full count (a value of FFFF Hex).

The decodes from each counter are logically "ORed" together and may be used as interrupt sources for the  $\overline{ELI}$  interrupt pin. Additionally the status of these bits may be observed by reading the Page Select Register (PSR), (see Section 8 for register details). In order to enable any of these threshold interrupts, the appropriate interrupt mask bit must be written to the Management and Interrupt Configuration Register; see Section 8 for register details.

In addition to their event masking functions, the Upper Event Counting Mask Register (UECMR) possesses two bits which control the operation of the counters. When written to a logic one, the reset on read bit "ROR" resets the counter after a processor read cycle is performed. If this operation is not selected, then in order to zero the counters, they must either be written with zeros by the processor, or allowed to roll over to all zeros. The freeze when full bit "FWF" prevents counter roll over by inhibiting count up cycles (these happen when chosen events occur), thus freezing the particular counter at FFFF Hex.

The port event counters may also be controlled by the Counter Decrement (CDEC) pin. As its name suggests, a

logic low state on this pin will decrement all the counters by a single value. The pulses on CDEC are internally synchronized and scheduled so as not to conflict with any "up counting" activity. If an up count and a down count occur simultaneously, then the down count is delayed until the up count has completed. This combination of up and down counting capability enables the RIC II's on-chip counters to provide a simple rolling average, or be used as extensions of larger off-chip counters.

**Note:** If the FWF option is enabled then the count down operation is disabled from those registers which have reached FFFF Hex and consequently have been frozen. Thus, if FWF is set and CDEC has been employed to provide a rate indication. A frozen counter indicates that a rate has been detected which has gone out of bounds, i.e., too fast increment or too slow increment. If the low count and high count decodes are employed as either interrupt sources or in a polling cycle, the direction of the rate excursion may be determined.

### NEW HUB MANAGEMENT COUNTERS

The RIC II adds 13 more 8-bit counters then provided on the DP83950 RIC. These counters will count events specified in the Event Count and Interrupt Mask Register 2 (ECIMR2), such as Frame Check Sequence, Frame Alignment Error, Partition, Out of Window Collision. This register also includes "Reset On Read" and "Freeze When Full" control bits.

It should be noted that Counter Decrement (CDEC) will not be used with the ECMR2. Also no real time or event logging interrupt, RTI or ELI, will be generated for this register.

### READING THE EVENT COUNTERS

The RIC II's external data bus is eight bits wide, since the event counters are 16 bits long, two processor read cycles are required to yield the counter value. In order to ensure that the read value is correct, and to allow simultaneous event counts with processor accesses, a temporary holding register is employed. A read cycle to either the lower or upper byte of a counter, causes both bytes to be latched into the holding register. Thus, when the other byte of the counter is obtained, the holding register is accessed, and not the actual counter register. This ensures that the upper and lower bytes contain the value sampled at the same instance in time, i.e., when the first read cycle to that counter occurred.

There is no restriction concerning whether the upper or lower byte is read first. However to ensure the "same instance value" is obtained, the reads of the upper then lower byte (or vice versa) should be performed as consecutive reads of the counter array. Other NON-COUNTER registers may be read in between these read cycles and also write cycles may be performed. If another counter is read, or the same byte of the original counter is read again, then the holding register is updated from the counter array, and the unread byte is lost.

If the reset on read option is employed, then the counter is reset after the transfer to the holding register is performed. Processor read and write cycles are scheduled in such a manner that they do not conflict with count up or count down operations. That is to say, in the case of a processor read, the count value is stable when it is loaded into the holding register. In the case of a processor write, the newly written value is stable so it may be incremented or decremented by any subsequent count operation. During the period the MLOAD pin is low, (power on reset) all counters are

## 6.0 Hub Management Support (Continued)

reset to zero and all count masks are forced into the disabled state. Section 8 of the data sheet details the address location of the port event counters.

### 6.2 EVENT RECORD FUNCTION

As previously stated each repeater port has its own Event Recording Register. This is an 8-bit status register. Each bit is dedicated to logging the occurrence of a particular event (see Section 8 for detailed description). The logging of these events is controlled by the Event Recording Mask Register, for an event to be recorded the particular mask bit must be set, (see Section 8 description of this register). Similar to the scheme employed for the event counters, the recorded events are latched during the repetition of a packet, and then automatically loaded into the recording registers at the end of transmission of a packet. When one of the unmasked events occurs, the particular port register bit is set. This status is visible to the user. All of the register bits for all of the ports are logically "ORed" together to produce a Flag Found "FF" signal. This indicator may be found by reading the Page Select Register. Additionally, an interrupt may be generated if the appropriate mask bit is enabled in the Management and Interrupt Configuration Register.

A processor read cycle to an Event Record Register resets any of the bits set in that register. Read operations are scheduled to guarantee non-changing data during a read cycle. Any internal bit setting event which immediately follows a processor read will be successful. The events which may be recorded are described below:

**Jabber Protection (JAB):** This flag goes active if the length of a received packet from the relevant port, causes the repeater state machine to enter the Jabber Protect state.

**Elasticity Buffer Error (ELBER):** This condition occurs if an Elasticity Buffer full or overflow occurs during packet reception. The flag is held inactive if a collision occurs during packet reception or if a phase lock error has already occurred during the repetition of the packet.

**Phase Lock Error (PLER):** A phase lock error is caused if the phase lock loop decoder loses lock during packet reception. Phase lock onto the received data stream may or may not be recovered later in the packet and data errors may have occurred. This flag is held inactive if a collision occurs.

**Non-SFD Packet (NSFD):** If a packet is received and the start of frame delimiter is not found, the flag will go active. The flag is held inactive if a collision occurs during packet repetition.

**Out of Window Collision (OWC):** The out of window collision flag for a port goes active when a collision is experienced outside of the network slot time.

**Partition (PART):** This flag goes active when a port becomes partitioned.

**Bad Link (BDLNK):** The flag goes active when a port is configured for 10BASE-T operation has entered the link lost state.

**Short Event reception (SE):** This flag goes active if the received packet is less than 74 bits long and no collision occurs during reception.

### 6.3 MANAGEMENT INTERFACE OPERATION

The Hub Management interface provides a mechanism to combine repeater status information with packet information to form a hub management status packet. The interface, a serial bus consisting of carrier sense, received clock and received data, is designed to connect one or multiple RIC II's over a backplane bus to a DP83932 "SONIC" network controller or DP83957 RIB. The SONIC/RIB and the RIC IIs form a powerful entity for network statistics gathering.

The interface consists of four pins:

|       |                                                                        |
|-------|------------------------------------------------------------------------|
| MRXC  | Management Receive Clock—10 MHz NRZ Clock output.                      |
| MCRS  | Management Carrier Sense—Input/Output indicating of valid data stream. |
| MRXD  | Management Receive Data—NRZ Data output synchronous to MRXC.           |
| PCOMP | Packet Compress—Input to truncate the packet's data field.             |

The first three signals mimic the interface between an Ethernet controller and a phase locked loop decoder (specifically the DP83932 SONIC and DP83910 SNI), these signals are driven by the RIC II receiving the packet. MRXC and MRXD compose an NRZ serial data stream compatible with the DP83932. The PCOMP signal is driven by logic on the processor board. The actual data stream transferred over MRXD is derived from data transferred over the IRD Inter-RIC bus line. These two data streams differ in two important characteristics:

1. At the end of packet repetition a hub management status field is appended to the data stream. This status field, consisting of 7 bytes, is shown in *Figures 6-1 and 6-2*. The information field is obtained from a number of packet status registers described below. In common with the 802.3 protocol the least significant bit of a byte is transmitted first.
2. While the data field of the repeated packet is being transferred over the management bus, received clock signals on the MRXC pin may be inhibited. This operation is under the control of the Packet Compress pin PCOMP. If PCOMP is asserted during repetition of the packet then MRXC signals are inhibited when the number of bytes (after SFD) transferred over the management bus equals the number indicated in the Packet Compress Decode Register. This register provides a means to delay the effect of the PCOMP signal, which may be generated early in the packet's repetition, until the desired moment. Packet compression may be used to reduce the amount of memory required to buffer packets when they are received and are waiting to be processed by hub management software. In this kind of application an address decoder, which forms part of the packet compress logic, would monitor the address fields as they are received over the management bus. If the destination address is not the address of the management node inside the hub, then packet compression could be employed. In this manner only the portion of the packet meaningful for hub management interrogation, i.e., the address fields, is transferred to the SONIC and is buffered in memory.

## 6.0 Hub Management Support (Continued)

If the repeated packet ends before  $\overline{\text{PCOMP}}$  is asserted or before the required number of bytes have been transferred, then the hub management status field is directly appended to the received data at a byte boundary. If the repeated packet is significantly longer than the value in the Decode Register requires, and  $\overline{\text{PCOMP}}$  is asserted, the status fields will be delayed until the end of packet repetition. During this delay period MRXC clocks are inhibited, but the MCRS signal remains asserted.

**Note:** If  $\overline{\text{PCOMP}}$  is asserted late in the packet, i.e., after the number of bytes defined by the packet compression register, then packet compression will not occur.

The Management Interface may be fine tuned to meet the timing considerations of the SONIC and the access time of its associated packet memory. This refinement may be performed in two ways:

1. The default mode of operation of the Management interface is to only transfer packets over the bus which have a start of frame delimiter. Thus "packets" that are only preamble/jam and do not convey any source or destination address information are inhibited. This filtering may be disabled by writing a logic zero to the Management Interface Configuration or "MIFCON" bit in the Management and Interrupt Configuration Register. See Section 8 for details.
2. The Management bus has been designed to accommodate situations of maximum network utilization, for example when collision generated fragments occur (these collision fragments may violate the IEEE802.3 IFG specification). The IFG required by the SONIC is a function of the time taken to release space in the receive FIFO and to perform end of packet processing (write status information into memory). These functions are primarily memory operations and consequently depend upon the bus latency and the memory access time of the system. In order to allow the system designer some discretion in choosing the speed of this memory, the RIC II may be configured to protect the SONIC from a potential FIFO overflow. This is performed by utilizing the Inter Frame Gap Threshold Select Register.

The value held in this register, plus one, defines, in network bit times, the minimum allowed gap between frames on the management bus. If the gap is smaller than this number then MCRS is asserted but MRXC clocks are inhibited. Consequently no data transfer is performed.

Thus the system designer may make the decision whether to gather statistics on all packets even if they occur with very small IFGs or to monitor a subset.

The status field, shown in *Figure 6-1*, contains information of six different types. They are contained in seven Packet Status Registers "PSRs":

1. The RIC II and port address fields [PSR(0) and (1)] can uniquely identify the repeater port receiving the packet out of a potential maximum of 832 ports sharing the same management bus (64 RIC IIs each with 13 ports). Thus all of the other status fields can be correctly attributed to the relevant port.

2. The status flags the RIC II produces for the event counters or recording latches are supplied with each packet [PSR(2)]. Additionally the clean receive CLN status is supplied to allow the user to determine the reliability of the address fields in the packet. The CLN status bit [PSR(1)] is set if no collisions are experienced during the repetition of the address fields.
3. The RIC II has an on-chip timer to indicate when, relative to the start of packet repetition, a collision, if any, occurred [PSR(3)]. There is also a timer which indicates how many bit times of IFG was seen on the network between repetition of this packet and the preceding one. This is provided by [PSR(6)].
4. If packet compression is employed, the receive byte count contained in the SONIC's packet descriptor will indicate the number of bytes transferred over the management bus rather than the number of bytes in the packet. For this reason the RIC II which receives the packet, counts the number of received bytes and transfers this over the management bus [PSR(4),(5)].
5. Appending a status field to a data packet will obviously result in a CRC error being flagged by the SONIC. For this reason the RIC II monitors the repeated data stream to check for CRC and FAE errors. In the case of FAE errors the RIC II provides additional dummy data bits, so that the status fields are always byte aligned.
6. As a final check upon the effectiveness of the management interface, the RIC II transfers a bus specific status bit to the SONIC. This flag Packet Compress Done PCOMP [PSR(0)], may be monitored by hub management software to check if the packet compression operation is enabled.

*Figure 6-3* shows an example of a packet being transmitted over the management bus. The first section of the diagram (moving from left to right) shows a short preamble and SFD pattern. The second region contains the packet's address and the start of the data fields. During this time logic on the processor/SONIC card would determine if packet compression should be used on this packet. The  $\overline{\text{PCOMP}}$  signal is asserted and packet transfer stops when the number of bytes transmitted equals the value defined in the decode register. Hence the MRXC signal is idle for the remainder of the packet's data and CRC fields. The final region shows the transfer of the RIC II's seven bytes of packet status.

The following pages describe these Hub Management registers which constitute the management status field.

Note that Packet Status Register 5 (PSR5) can be configured to remain identical in the RIC II as in the RIC, or PSR5 can be modified to include the RUNT and SAM (source address mismatch) information. PSR5 register bit allocation is determined by the value of bit D2, MPS (Modify Packet Status), in the Global Security Register. When the MPS bit is set, PSR5 register is modified.

## 6.0 Hub Management Support (Continued)

| Packet Status Register (PSR) (Note 1) |                     | D7    | D6    | D5    | D4    | D3    | D2    | D1     | D0   |
|---------------------------------------|---------------------|-------|-------|-------|-------|-------|-------|--------|------|
| PSR(0)                                |                     | A5    | A4    | A3    | A2    | A1    | A0    | PCOMPD | resv |
| PSR(1)                                |                     | CRCER | FAE   | COL   | CLN   | PA3   | PA2   | PA1    | PA0  |
| PSR(2)                                |                     | SE    | OWC   | NSFD  | PLER  | ELBER | JAB   | CBT9   | CBT8 |
| PSR(3)<br>Collision Bit Timer         |                     | CBT7  | CBT6  | CBT5  | CBT4  | CBT3  | CBT2  | CBT1   | CBT0 |
| PSR(4)<br>Lower Repeat Byte Count     |                     | RBY7  | RBY6  | RBY5  | RBY4  | RBY3  | RBY2  | RBY1   | RBY0 |
| PSR(5)<br>Upper Repeat<br>Byte Count  | MPS = 0<br>(Note 2) | RBY15 | RBY14 | RBY13 | RBY12 | RBY11 | RBY10 | RBY9   | RBY8 |
|                                       | MPS = 1             | res   | res   | res   | SAM   | RUNT  | RBY10 | RBY9   | RBY9 |
| PSR(6)<br>Inter Frame Gap Bit Timer   |                     | IBT7  | IBT6  | IBT5  | IBT4  | IBT3  | IBT2  | IBT1   | IBT0 |

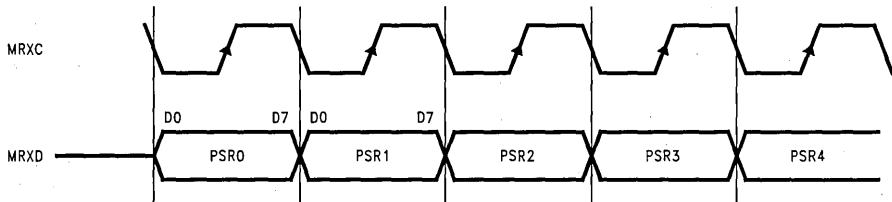
**Note 1:** These registers may only be reliably accessed via the management interface. Due to the nature of these registers they may not be accessed (read or write cycles) via the processor interface.

**Note 2:** When MPS (Modify Packet Status) bit in the Global Security Register is:

MPS=0, Do not modify Packet Status Register 5. The RIC II PSR5 is the same as the RIC PSR5.

MPS=1, The PSR5 register is modified in the RIC II.

**FIGURE 6-1. Hub Management Status Field**



**FIGURE 6-2. Management Bus Packet Status Register Timing**

TL/F/12499-19

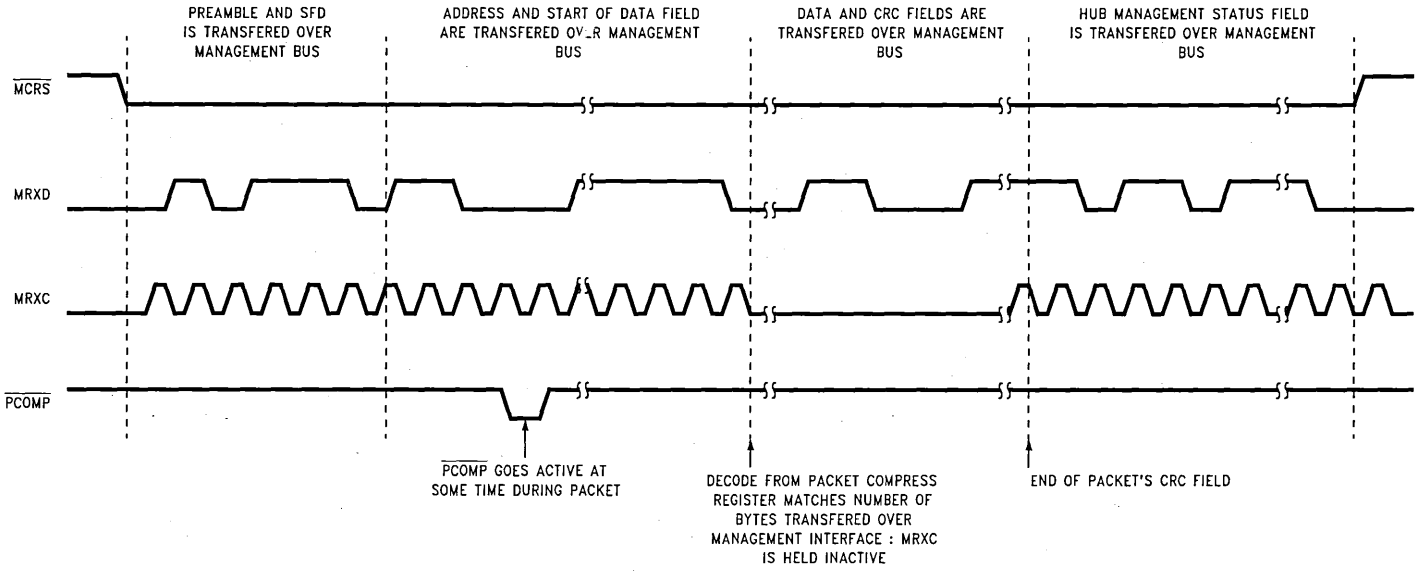


FIGURE 6-3. Operation of the Management Bus

TL/F/12499-20

3-125

## 6.0 Hub Management Support (Continued)

### PACKET STATUS REGISTER 0

|    |    |    |    |    |    |        |      |
|----|----|----|----|----|----|--------|------|
| D7 | D6 | D5 | D4 | D3 | D2 | D1     | D0   |
| A5 | A4 | A3 | A2 | A1 | A0 | PCOMPD | resv |

| Bit    | Symbol | Description                                                                                                                                                                                                                  |
|--------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0     | resv   | <b>RESERVED FOR FUTURE USE:</b> This bit is currently undefined management software should not examine the state of this bit.                                                                                                |
| D1     | PCOMPD | <b>PACKET COMPRESSION DONE:</b> If packet compression is utilized, this bit informs the user that compression was performed, i.e., the packet was long enough to require compression.                                        |
| D(7:2) | A(5:0) | <b>RIC II ADDRESS (5:0):</b> This address is defined by the user and is supplied when writing to the RIC II Address Register. It is used by hub management software to distinguish between RIC IIs in a multi-RIC II system. |

### PACKET STATUS REGISTER 1

|        |     |     |     |     |     |     |     |
|--------|-----|-----|-----|-----|-----|-----|-----|
| D7     | D6  | D5  | D4  | D3  | D2  | D1  | D0  |
| CR CER | FAE | COL | CLN | PA3 | PA2 | PA1 | PA0 |

| Bit    | Symbol  | Description                                                                                                                                                                                                    |
|--------|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D(3:0) | PA(3:0) | <b>PORT ADDRESS:</b> This field defines the port which is receiving the packet.                                                                                                                                |
| D4     | CLN     | <b>CLEAN RECEIVE:</b> This bit is asserted provided no collision activity occurs during repetition of the source and destination address fields, and the packet is of sufficient size to contain these fields. |
| D5     | COL     | <b>COLLISION:</b> If a receive or transmit collision occurs during packet repetition the collision bit is asserted.                                                                                            |
| D6     | FAE     | <b>FRAME ALIGNMENT ERROR:</b> This bit is asserted if a Frame Alignment Error occurred in the repeated packet.                                                                                                 |
| D7     | CR CER  | <b>CRC ERROR:</b> This bit is asserted if a CRC Error occurred in the repeated packet.<br>This status flag should not be tested if the COL bit is asserted since the error may be simply due to the collision. |

### PACKET STATUS REGISTER 2

|    |     |      |      |       |     |      |      |
|----|-----|------|------|-------|-----|------|------|
| D7 | D6  | D5   | D4   | D3    | D2  | D1   | D0   |
| SE | OWC | NSFD | PLER | ELBER | JAB | CBT9 | CBT8 |

| Bit    | Symbol  | Description                                                                                                                                                                                                                                       |
|--------|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D(1:0) | CT(9:8) | <b>COLLISION TIMER BITS 9 AND 8:</b> These two bits are the upper bits of the collision bit timer.                                                                                                                                                |
| D2     | JAB     | <b>JABBER EVENT:</b> This bit indicates that the receive packet was so long the repeater was forced to go into a jabber protect condition.                                                                                                        |
| D3     | ELBER   | <b>ELASTICITY BUFFER ERROR:</b> During the packet an Elasticity Buffer under/overflow occurred.                                                                                                                                                   |
| D4     | CR CER  | <b>CARRIER ERROR EVENT:</b> The packet suffered sufficient jitter/noise corruption to cause the phase lock loop decoder to loose lock.                                                                                                            |
| D5     | NSFD    | <b>NON-SFD:</b> The repeated packet did not contain a Start of Frame Delimiter. When this bit is set the Repeat Byte Counter counts the length of the entire packet. When this bit is not set the byte counter only counts post SFD bytes. (Note) |
| D6     | OWC     | <b>OUT OF WINDOW COLLISION:</b> The packet suffered an out of window collision.                                                                                                                                                                   |
| D7     | SE      | <b>SHORT EVENT:</b> The received activity was so small it met the criteria to be classed as a short event.                                                                                                                                        |

**Note:** The operation of this bit is not inhibited by the occurrence of a collision during packet repetition (see description of the Repeat Byte Counter below).

## 6.0 Hub Management Support (Continued)

### MODIFIED PACKET STATUS REGISTER 5 (MPS = 1 IN GSR REGISTER)

RIC II provides an option for a new Packet Status Register 5 (PSR5) field. On the seven bytes of management status field, PSR5 has been modified to indicate the source address mismatch information (SAM bit) for security purposes.

By using this option, the maximum received byte count changes to 2048 (2<sup>11</sup>). As soon as the counter reaches this number, it will freeze, instead of rolling over and starting again on the reception of the next packet.

A RUNT bit has also been added to this register indicating whether the last packet received by a port was RUNT. (A packet is RUNT when its length is greater than or equal to Short Event and less than or equal to 64 bytes from SFD.)

The other registers comprise the remainder of the collision timer register [PSR(3)], the Repeat Byte Count registers [PSR(4) and PSR(5)], and the Inter Frame Gap Counter "IFG" register [PSR(6)].

#### COLLISION BIT TIMER

The Collision Timer counts in bit times the time between the start of repetition of the packet and the detection of the packet's first collision. The Collision counter increments as the packet is repeated and freezes when a collision occurs. The value in the counter is only valid when the collision bit "COL" in [PSR(1)] is set.

#### REPEAT BYTE COUNTER

The Repeat Byte Counter is a 16-bit counter which can perform two functions. In cases where the transmitted packet possesses an SFD, the byte counter counts the number of received bytes after the SFD field. Alternatively, if no SFD is repeated, the counter reflects the length of the packet (counted in bytes) starting at the beginning of the preamble field. When performing the latter function, the counter is

shortened to 7 bits when MPS = 0 in the GSR register. Thus, the maximum count value is 127 bytes. The counter is shortened to 11 bits when MPS = 1 in the GSR register. In this configuration, the maximum received byte count changes to 2048 bytes. The mode of counting is indicated by the "NSFD" bit in [PSR(2)]. In order to check if the received packet was genuinely a Non-SFD packet, the status of the COL bit should be checked. During collisions SFD fields may be lost or created, Management software should be robust to this kind of behavior.

#### INTER FRAME GAP (IFG) BIT TIMER

The IFG counter counts in bit times the period in between repeater transmissions. The IFG counter increments whenever the RIC II is not transmitting a packet. If the IFG is long, i.e., greater than 255 bits the counter sticks at this value. Thus an apparent count value of 255 should be interpreted as 255 or more bit times.

#### 6.4 DESCRIPTION OF HARDWARE CONNECTION FOR MANAGEMENT INTERFACE

The RIC II has been designed so it may be connected to the Management bus directly or via external bus transceivers. The latter is advantageous in large repeaters. In this application the system backplane is often heavily loaded beyond the drive capabilities of the on-chip bus drivers.

The uni-directional nature of information transfer on the MCRS, MRXD and MRXC signals, means a single open drain output pin is adequate for each of these signals. The Management Enable (MEN) RIC II output pin performs the function of a drive enable for an external bus transceiver if one is required.

In common with the Inter-RIC bus signals ACTN, ANYXN, COLN and IRE the MCRS active level asserted by the MCRS output is determined by the state of the BINV Mode Load configuration bit.

### MODIFIED PACKET STATUS REGISTER 5 (MPS = 1 IN GSR REGISTER)

| D7  | D6  | D5  | D4  | D3   | D2    | D1   | D0   |
|-----|-----|-----|-----|------|-------|------|------|
| res | res | res | SAM | RUNT | RBY10 | RBY9 | RBY8 |

| Bit    | R/W | Symbol | Description                                                                                                                                                                                             |
|--------|-----|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0     | NA  | RBY8   | Eighth bit of receive byte count.                                                                                                                                                                       |
| D1     | NA  | RBY9   | Ninth bit of receive byte count.                                                                                                                                                                        |
| D2     | NA  | RBY10  | Tenth bit of receive byte count.                                                                                                                                                                        |
| D3     | NA  | RUNT   | <b>RUNT:</b> A packet whose length is less than or equal to 64 bytes from SFD and greater than or equal to SE length.<br>0: Last packet received was not a runt.<br>1: Last packet received was a runt. |
| D4     | NA  | SAM    | <b>SOURCE ADDRESS MISMATCH:</b><br>0: Source address match occurred for the last packet.<br>1: Source address mismatch occurred for the last packet.                                                    |
| D[7:5] | NA  | res    | <b>RESERVED FOR FUTURE USE:</b> Reads as a logic 0.                                                                                                                                                     |

## 7.0 Port Block Functions

The RIC II has 13 port logic blocks (one for each network connection). In addition to the packet repetition operations already described, the port block performs two other functions:

1. the physical connection to the network segment (transceiver function).
2. it provides a means to protect the network from malfunctioning segments (segment partition).

Each port has its own status register. This register allows the user to determine the current status of the port and configure a number of port specific functions.

### 7.1 TRANSCEIVER FUNCTIONS

The RIC II may connect to network segments in three ways:

1. over AUI cable to transceiver boxes,
2. directly to board mounted transceivers,
3. to twisted pair cable via a simple interface.

The first method is only supported by RIC II port 1 (the AUI port). Options (2) and (3) are available on ports 2 to 13. The selection of the desired option is made at device initialization during the Mode Load operation. The Transceiver Bypass XBYPAS configuration bits are used to determine whether the ports will utilize the on-chip 10BASE-T transceivers, or bypass these in favor of external transceivers. Four possible combinations of port utilization are supported:

All ports (2 to 13) use the external Transceiver Interface.

Ports 2 to 5 use the external interface, 6 to 13 use the internal 10BASE-T transceivers.

Ports 2 to 7 use the external interface, 8 to 13 use the internal 10BASE-T transceivers.

All ports (2 to 13) use the internal 10BASE-T transceivers.

### 10BASE-T TRANSCEIVER OPERATION

The RIC II contains virtually all the digital and analog circuits required for connection to 10BASE-T network segments. The only additional active component is an external driver package. The connection for a RIC II port to a 10BASE-T segment is shown in *Figure 7-1*. The diagram shows the components required to connect one of the RIC II's ports to a 10BASE-T segment. The major components are the driver package, a member of the 74ACT family, and an integrated filter/choke network.

The operation of the 10BASE-T transceiver's logical functions may be modified by software control. The default mode of operation is for the transceivers to transmit and expect reception of link pulses. This may be modified if a logic one is written to the  $\overline{\text{GDLNK}}$  bit of a port's status register. The port's transceiver will operate normally but will not transmit link pulses nor monitor their reception. Thus the entry to a link fail state and the associated modification of transceiver operation will not occur.

The on-chip 10BASE-T transceivers automatically detect and correct the polarity of the received data stream. This polarity detection scheme relies upon the polarity of the received link pulses and the end of packet waveform. Polarity detection and correction may be disabled under software control.

### EXTERNAL TRANSCEIVER OPERATION

RIC II ports 2 to 13 may be connected to media other than twisted-pair by opting to bypass the on-chip transceivers. When using external transceivers the user must have the external transceivers perform collision detection and the

other functions associated with an IEEE 802.2 Media Access Unit. *Figure 7-2* shows the connection between a repeater port and a coaxial transceiver using the AUI type interface.

### 7.2 SEGMENT PARTITION

Each of the RIC II's ports has a dedicated state machine to perform the functions defined by the IEEE partition algorithm as shown in *Figure 7-3*. To allow users to customize this algorithm for different applications, a number of user selected options are available during device configuration at power up (the Mode Load cycle).

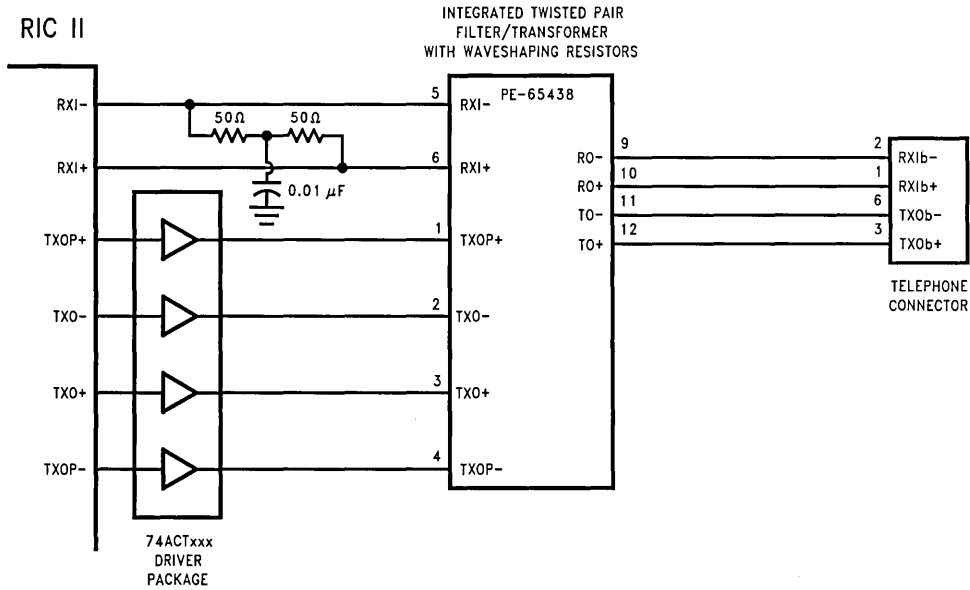
Five different options are provided:

1. Operation of the 13 partition state machines may be disabled via the disable partition  $\overline{\text{DPART}}$  configuration bit (pin D6).
2. The value of consecutive counts required to partition a segment (the  $\overline{\text{CCLimit}}$  specification) may be set at either 31 or 63 consecutive collisions.
3. The use of the TW5 specification in the partition algorithm differentiates between collisions which occur early in a packet (before TW5 has elapsed) and those which occur late in the packet (after TW5 has elapsed). These late or "out of window" collisions can be regarded in the same manner as early collisions if the Out of Window Collision Enable  $\overline{\text{OWCE}}$  option is selected. This configuration bit is applied to the D4 pin during the Mode Load operation. The use of  $\overline{\text{OWCE}}$  delays until the end of the packet the operation of the state diagram branch marked (1) and enables the branch marked (2) in *Figure 7-3*.
4. The operation of the ports' state machines when reconnecting a segment may also be modified by the user. The Transmit Only  $\overline{\text{TXONLY}}$  configuration bit allows the user to prevent segment reconnection unless the reconnecting packet is being sourced by the repeater. In this case the repeater is transmitting on to the segment, rather than the segment transmitting when the repeater is idle. The normal mode of reconnection does not differentiate between such packets. The  $\overline{\text{TXONLY}}$  configuration bit is input on pin D5 during the Mode Load cycle. If this option is selected the operation of the state machine branch marked (3) in *Figure 7-3* is affected.
5. The RIC II may be configured to use an additional criterion for segment partition. This is referred to as loop back partition. If this operation is selected the partition state machine monitors the receive and collision inputs from a network segment to discover if they are active when the port is transmitting. Thus determining if the network transceiver is looping back the data pattern from the cable. A port may be partitioned if no data or collision signals are seen by the partition logic in the following window: 61 to 96 network bit times after the start of transmission; see data sheet Section 8 for details. A segment partitioned by this operation may be reconnected in the normal manner.

In addition to the autonomous operation of the partition state machines, the user may reset these state machines. This may be done individually to each port by writing a logic one to the  $\overline{\text{PART}}$  bit in its status register. The port's partition state machine and associated counters are reset and the port is reconnected to the network. The reason why a port became partitioned may be discovered by the user by reading the port's status register.



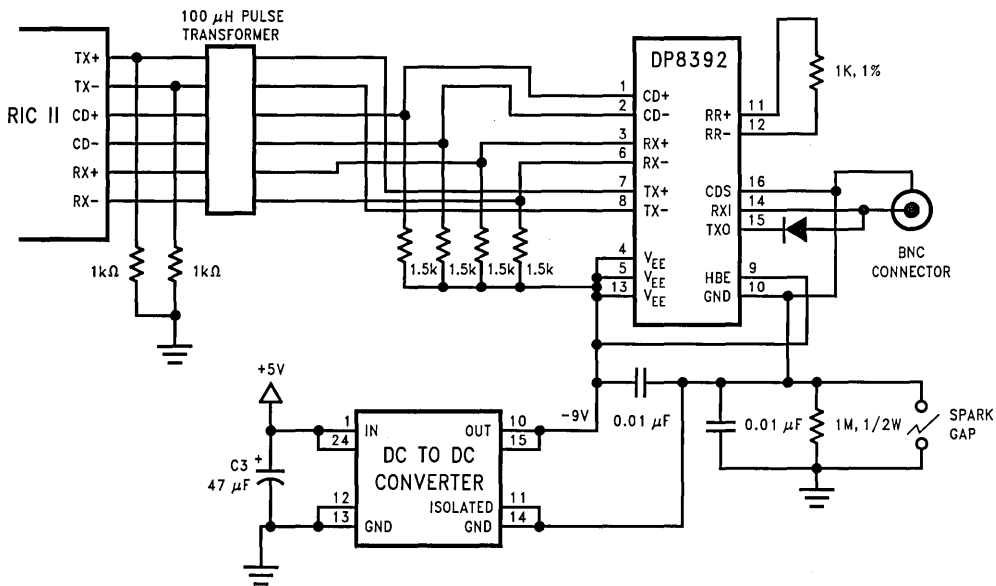
### 7.0 Port Block Functions (Continued)



TL/W/12499-21

**Note:** For recommended modules, see "Ethernet Magnetics Vendors for 10BASE-T, 10BASE2, and BASE5". In this example, Pulse Engineering's PE-65438 device is used.

**FIGURE 7-1. Port Connection to a 10-BASE-T Segment**



TL/W/12499-22

**FIGURE 7-2. Port Connection to a 10-BASE2 Segment (AUI Interface Selected)**

The above diagrams show a RIC II port (numbers 2 to 13) connected to a 10BASE-T and a 10BASE2 segment. The values of any components not indicated above are to be determined.

## 7.0 Port Block Functions (Continued)

### 7.3 PORT STATUS REGISTER FUNCTIONS

Each RIC II port has its own status register. In addition to providing status concerning the port and its network segment, the register allows the following operations to be performed upon the port:

1. Port Disable
2. Link Disable
3. Partition Reconnection
4. Selection between normal and reduced squelch levels

Note that the link disable and port disable functions are mutually exclusive functions, i.e. disabling link does not affect receiving and transmitting from/to that port and disabling a port does not disable link.

When a port is disabled packet transmission and reception between the port's segment and the rest of the network is prevented.

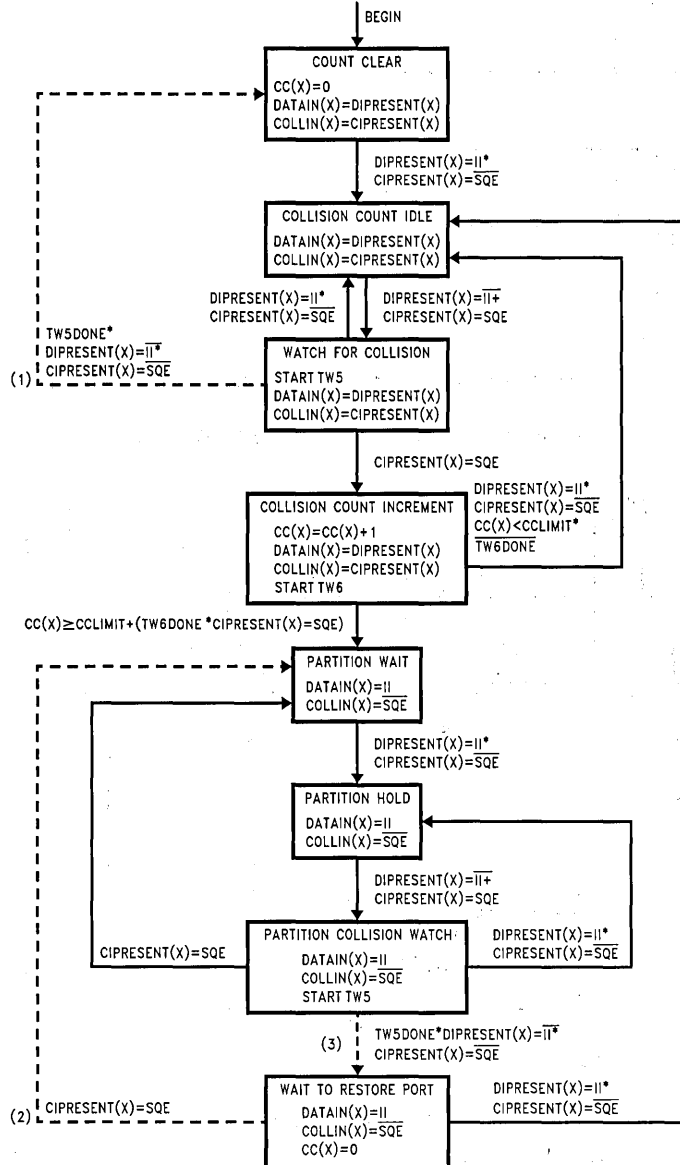


FIGURE 7-3. IEEE Segment Partition Algorithm

TL/W/12499-23

## 7.0 Port Block Functions (Continued)

### 7.4 LOCAL PORTS AND INTER-RIC BUS EXPECTED ACTIVITY

The RIC II incorporates security options into the repeater. The configuration of the security features can be performed globally, or on a per port basis. Upon packet reception by the RIC II, depending on port configuration, the repeater will either: transmit the actual data intact to the port, or transmit pseudo random data to the port during the data field of the packet.

RIC II security features can be globally enabled/disabled during the MLOAD process, or via the RIC II Configuration Register bit GSE (global security enable). When GSE is set, the device will, for all ports, set the port SME (Security Mode), ESA (Source Address Security), EDA (Destination Address Security), LME (Learn Mode Enable) bits in the Port Security Configuration Register (PSCR).

Learning Mode for all the port CAMs can be globally enabled during the MLOAD process, or via the GLME (Global Learn Mode Enable) bit in the Global Security Register (GSR). When GLME is set, the device will, for all ports, set the LME (Learn Mode Enable) bit in the PSCR register.

In a multi RIC II repeater environment, each RIC II will get the packet intact over the Inter-RIC bus (except on source address mismatch occurrence when configured in the security mode). Each RIC II will transmit either the real data or pseudo random data to the port depending on the port configuration.

#### Rule of Thumb (See table on next page)

##### 1. Security Mode Disabled

RIC II performs the same repeater operations as the RIC. The received data is transmitted to all ports, and on the Inter-RIC bus.

##### 2. Security Mode Enabled

###### a. When a port's ESA = 0,

- 1) and the port's EDA = 0, then the repeater will repeat the data on the port, and the Inter-RIC bus.
- 2) and the port's EDA = 1, then the repeater will repeat the data on destination address match. On a destination address mismatch, the repeater will transmit random data on that port. In both cases, the repeater will transmit data on the Inter-RIC bus.

###### b. When a port's ESA = 1,

- 1) and the port's EDA = 0, then on a valid source address match, the repeater will repeat the data on that port, and on the Inter-RIC bus. If source address mismatch occurs, then the repeater will transmit random data to the port, and on the Inter-RIC bus.
- 2) and the port's EDA = 1, then on a valid source and destination address match, the repeater will repeat the data on the port. If source address matches, but the destination address does not match, then the repeater will transmit random data to that port. In both of these cases, the repeater will repeat the data on the Inter-RIC bus. When source address mismatch occurs, then the repeater will transmit random data to the port and on the Inter-RIC bus.

The following table describes the type of data in the packet (actual data or pseudo random data) is transmitted out of the ports, and over the Inter-RIC bus. It is assumed that the repeater is powered up in security mode (GLME = 0).

For example, suppose the repeater is in security mode (SME = 1), and configured to perform address comparison only on destination address (ESA = 0 and EDA = 1). If a packet is received whose destination address does not match with that stored address in a designated CAM, then all the transmitting ports switch to random packet, while the data is transmitted intact over the Inter-RIC bus. The other cascaded repeaters will compare the packet's destination address with their own internal CAMs for proper decision making.

## 7.0 Port Block Functions (Continued)

### 7.5 LOCAL PORTS AND INTER-RIC BUS DATA FIELD CONTENTS

| SME      | ESA      | EDA | Source Address of Packet | Destination Address of Packet | Transmitting Ports | Inter-RIC Bus |
|----------|----------|-----|--------------------------|-------------------------------|--------------------|---------------|
| 0        | X        | X   | X                        | X                             | Repeat             | Repeat        |
| 1        | 0        | 0   | X                        | X                             | Repeat             | Repeat        |
|          |          | 1   | Match                    | Match                         | Repeat             | Repeat        |
|          |          |     | Match                    | Mismatch                      | Random             | Repeat        |
|          |          |     | Mismatch                 | Match                         | Repeat             | Repeat        |
|          | Mismatch |     | Mismatch                 | Random                        | Repeat             |               |
|          | 1        | 0   | Match                    | Match                         | Repeat             | Repeat        |
|          |          |     | Match                    | Mismatch                      | Repeat             | Repeat        |
|          |          |     | Mismatch                 | X                             | Random             | Random        |
|          |          | 1   | Match                    | Match                         | Repeat             | Repeat        |
|          |          |     | Match                    | Mismatch                      | Random             | Repeat        |
| Mismatch |          |     | X                        | Random                        | Random             |               |

Note: SME: Security Mode bit in the Port Security Configuration Register (PSCR).

ESA: Source Address Security bit in the PSCR register.

EDA: Destination Address Security bit in the PSCR register.

## 8.0 RIC II Registers

### RIC II REGISTER ADDRESS MAP

The RIC II's registers may be accessed by applying the required address to the five Register Address (RA(4:0)) input pins. Pin RA4 makes the selection between the upper and lower halves of the register array. The lower half of the register map consists of 16 registers:

- 1 RIC II Real Time Status and Configuration register,
- 13 Port Real Time Status registers,
- 1 RIC II Configuration register,
- 1 Real Time Interrupt Status register.

These registers may be directly accessed at any time via the RA(4:0) pins, (RA4 = 0).

The upper half of the register map, (RA4 = 1), is organized as 15 pages of registers. These pages include registers for port security configuration (global and on a per port basis), event count registers, port CAM and shared CAM locations, CAM location mask registers, etc. See Memory Map and Register Description sections for details.

Register access within these pages is performed using the RA(4:0) pins, (RA4 = 1). Page switching is performed by writing to the Page Selection bits (PSEL3,2,1, and 0). These bits are found in the Page Select Register, located at address 10 hex on each page of the upper half of the register array. At power on these bits default to 0 Hex, i.e., page zero.

On the RIC II the following registers have been added/modified from the RIC registers:

1. Page Select Register
2. ECIMR-2 register added to page (0)
3. GSR register added to page (0)
4. Upper and Lower EIR registers added to page (1)
5. Added all registers on pages (4)-(15)
6. Modification Option for Management Packet Status Register 5 (PSR5) on the Management Bus.

## 8.0 RIC II Registers (Continued)

Register Address Map

| Name    |                                             |                                    |                                         |                   |
|---------|---------------------------------------------|------------------------------------|-----------------------------------------|-------------------|
| Address | PAGE (0)                                    | PAGE (1)                           | PAGE (2)                                | PAGE (3)          |
| 00H     | RIC II Status and Configuration Register    |                                    |                                         |                   |
| 01H     | Port 1 Status Register                      |                                    |                                         |                   |
| 02H     | Port 2 Status Register                      |                                    |                                         |                   |
| 03H     | Port 3 Status Register                      |                                    |                                         |                   |
| 04H     | Port 4 Status Register                      |                                    |                                         |                   |
| 05H     | Port 5 Status Register                      |                                    |                                         |                   |
| 06H     | Port 6 Status Register                      |                                    |                                         |                   |
| 07H     | Port 7 Status Register                      |                                    |                                         |                   |
| 08H     | Port 8 Status Register                      |                                    |                                         |                   |
| 09H     | Port 9 Status Register                      |                                    |                                         |                   |
| 0AH     | Port 10 Status Register                     |                                    |                                         |                   |
| 0BH     | Port 11 Status Register                     |                                    |                                         |                   |
| 0CH     | Port 12 Status Register                     |                                    |                                         |                   |
| 0DH     | Port 13 Status Register                     |                                    |                                         |                   |
| 0EH     | RIC II Configuration Register               |                                    |                                         |                   |
| 0FH     | Real Time Interrupt Register                |                                    |                                         |                   |
| 10H     | Page Select Register                        |                                    |                                         |                   |
| 11H     | Device Type Register                        | Port 1 Event Record Register (ERR) | res                                     | res               |
| 12H     | Lower Event Count Mask Register (ECMR)      | Port 2 ERR                         | Port 1 Lower Event Count Register (ECR) | Port 8 Lower ECR  |
| 13H     | Upper ECMR                                  | Port 3 ERR                         | Port 1 Upper ECR                        | Port 8 Upper ECR  |
| 14H     | Event Record Mask Register                  | Port 4 ERR                         | Port 2 Lower ECR                        | Port 9 Lower ECR  |
| 15H     | <b>ECIMR - 2</b>                            | Port 5 ERR                         | Port 2 Upper ECR                        | Port 9 Upper ECR  |
| 16H     | Management/Interrupt Configuration Register | Port 6 ERR                         | Port 3 Lower ECR                        | Port 10 Lower ECR |
| 17H     | RIC II Address Register                     | Port 7 ERR                         | Port 3 Upper ECR                        | Port 10 Upper ECR |
| 18H     | Packet Compress Decode Register             | Port 8 ERR                         | Port 4 Lower ECR                        | Port 11 Lower ECR |
| 19H     | res                                         | Port 9 ERR                         | Port 4 Upper ECR                        | Port 11 Upper ECR |
| 1AH     | res                                         | Port 10 ERR                        | Port 5 Lower ECR                        | Port 12 Lower ECR |
| 1BH     | res                                         | Port 11 ERR                        | Port 5 Upper ECR                        | Port 12 Upper ECR |
| 1CH     | res                                         | Port 12 ERR                        | Port 6 Lower ECR                        | Port 13 Lower ECR |
| 1DH     | <b>GSR</b>                                  | Port 13 ERR                        | Port 6 Upper ECR                        | Port 13 Upper ECR |
| 1EH     | res                                         | <b>Upper EIR</b>                   | Port 7 Lower ECR                        | res               |
| 1FH     | IFG Threshold Select                        | <b>Lower EIR</b>                   | Port 7 Upper ECR                        | res               |

Note: Registers written in **bold** are the new RIC II specific registers which are not present in the RIC.

## 8.0 RIC II Registers (Continued)

### Register Address Map (Continued)

| Name    |               |              |              |               |
|---------|---------------|--------------|--------------|---------------|
| Address | PAGE (4)      | PAGE (5)     | PAGE (6)     | PAGE (8)      |
| 11H     | Port 1 ECR-2  | Port 1 CAM 1 | Port 5 PCPR  | Port 9 PSCR   |
| 12H     | Port 2 ECR-2  | Port 1 CAM 2 | Port 5 CAM 1 | Port 9 PCPR   |
| 13H     | Port 3 ECR-2  | Port 2 PSCR  | Port 5 CAM 2 | Port 9 CAM 1  |
| 14H     | Port 4 ECR-2  | Port 2 PCPR  | Port 6 PSCR  | Port 9 CAM 2  |
| 15H     | Port 5 ECR-2  | Port 2 CAM 1 | Port 6 PCPR  | Port 10 PSCR  |
| 16H     | Port 6 ECR-2  | Port 2 CAM 2 | Port 6 CAM 1 | Port 10 PCPR  |
| 17H     | Port 7 ECR-2  | Port 3 PSCR  | Port 6 CAM 2 | Port 10 CAM 1 |
| 18H     | Port 8 ECR-2  | Port 3 PCPR  | Port 7 PSCR  | Port 10 CAM 2 |
| 19H     | Port 9 ECR-2  | Port 3 CAM 1 | Port 7 PCPR  | Port 11 PSCR  |
| 1AH     | Port 10 ECR-2 | Port 3 CAM 2 | Port 7 CAM 1 | Port 11 PCPR  |
| 1BH     | Port 11 ECR-2 | Port 4 PSCR  | Port 7 CAM 2 | Port 11 CAM 1 |
| 1CH     | Port 12 ECR-2 | Port 4 PCPR  | Port 8 PSCR  | Port 11 CAM 2 |
| 1DH     | Port 13 ECR-2 | Port 4 CAM 1 | Port 8 PCPR  | Port 12 PSCR  |
| 1EH     | Port 1 PSCR   | Port 4 CAM 2 | Port 8 CAM 1 | Port 12 PCPR  |
| 1FH     | Port 1 PCPR   | Port 5 PSCR  | Port 8 CAM 2 | Port 12 CAM 1 |

| Name    |               |               |                |                |
|---------|---------------|---------------|----------------|----------------|
| Address | PAGE (9)      | PAGE (10)     | PAGE (11)      | PAGE (12)      |
| 11H     | Port 12 CAM 2 | SCAM Lo 3     | SCAM Lo 8      | SCAM Lo 13     |
| 12H     | Port 13 PSCR  | CLMR Lo Loc 3 | CLMR Lo Loc 8  | CLMR Lo Loc 13 |
| 13H     | Port 13 PCPR  | CLMR Hi Loc 3 | CLMR Hi Loc 8  | CLMR Hi Loc 13 |
| 14H     | Port 13 CAM 1 | SCAM Lo 4     | SCAM Lo 9      | SCAM Lo 14     |
| 15H     | Port 13 CAM 2 | CLMR Lo Loc 4 | CLMR Lo Loc 9  | CLMR Lo Loc 14 |
| 16H     | SCVR 1        | CLMR Hi Loc 4 | CLMR Hi Loc 9  | CLMR Hi Loc 14 |
| 17H     | SCVR 2        | SCAM Lo 5     | SCAM Lo 10     | SCAM Lo 15     |
| 18H     | SCVR 3        | CLMR Lo Loc 5 | CLMR Lo Loc 10 | CLMR Lo Loc 15 |
| 19H     | SCVR 4        | CLMR Hi Loc 5 | CLMR Hi Loc 10 | CLMR Hi Loc 15 |
| 1AH     | SCAM Lo 1     | SCAM Lo 6     | SCAM Lo 11     | SCAM Lo 16     |
| 1BH     | CLMR Lo Loc 1 | CLMR Lo Loc 6 | CLMR Lo Loc 11 | CLMR Lo Loc 16 |
| 1CH     | CLMR Hi Loc 1 | CLMR Hi Loc 6 | CLMR Hi Loc 11 | CLMR Hi Loc 16 |
| 1DH     | SCAM Lo 2     | SCAM Lo 7     | SCAM Lo 12     | SCAM Lo 17     |
| 1EH     | CLMR Lo Loc 2 | CLMR Lo Loc 7 | CLMR Lo Loc 12 | CLMR Lo Loc 17 |
| 1FH     | CLMR Hi Loc 2 | CLMR Hi Loc 7 | CLMR Hi Loc 12 | CLMR Hi Loc 17 |

**8.0 RIC II Registers** (Continued)**Register Address Map** (Continued)**Name**

| <b>Address</b> | <b>PAGE (13)</b> | <b>PAGE (14)</b> | <b>PAGE (15)</b> |
|----------------|------------------|------------------|------------------|
| 11H            | SCAM Lo 18       | SCAM Lo 23       | SCAM Lo 28       |
| 12H            | CLMR Lo Loc 18   | CLMR Lo Loc 23   | CLMR Lo Loc 28   |
| 13H            | CLMR Hi Loc 18   | CLMR Hi Loc 23   | CLMR Hi Loc 28   |
| 14H            | SCAM Lo 19       | SCAM Lo 24       | SCAM Lo 29       |
| 15H            | CLMR Lo Loc 19   | CLMR Lo Loc 24   | CLMR Lo Loc 29   |
| 16H            | CLMR Hi Loc 19   | CLMR Hi Loc 24   | CLMR Hi Loc 20   |
| 17H            | SCAM Lo 20       | SCAM Lo 25       | SCAM Lo 30       |
| 18H            | CLMR Lo Loc 20   | CLMR Lo Loc 25   | CLMR Lo Loc 30   |
| 19H            | CLMR Hi Loc 20   | CLMR Hi Loc 25   | CLMR Hi Loc 30   |
| 1AH            | SCAM Lo 21       | SCAM Lo 26       | SCAM Lo 31       |
| 1BH            | CLMR Lo Loc 21   | CLMR Lo Loc 26   | CLMR Lo Loc 31   |
| 1CH            | CLMR Hi Loc 21   | CLMR Hi Loc 26   | CLMR Hi Loc 31   |
| 1DH            | SCAM Lo 22       | SCAM Lo 27       | SCAM Lo 32       |
| 1EH            | CLMR Lo Loc 22   | CLMR Lo Loc 27   | CLMR Lo Loc 32   |
| 1FH            | CLMR Hi Loc 22   | CLMR Hi Loc 27   | CLMR Hi Loc 32   |

## 8.0 RIC II Registers (Continued)

Register Array Bit Map Addresses 00H to 10H

| Address (Hex) | D7     | D6     | D5     | D4     | D3     | D2    | D1    | D0    |
|---------------|--------|--------|--------|--------|--------|-------|-------|-------|
| 00            | BINV   | BYPAS2 | BYPAS1 | APART  | JAB    | AREC  | ACOL  | resv  |
| 01 to 0D      | DISPT  | SQL    | PTYPE1 | PTYPE0 | PART   | REC   | COL   | GDLNK |
| 0E            | MINMAX | DPART  | TXONLY | OWCE   | LPPART | CLIM  | TW2   | GSE   |
| 0F            | IVCTR3 | IVCTR2 | IVCTR1 | IVCTR0 | ISRC3  | ISRC2 | ISRC1 | ISRC0 |
| 10            | FC     | HC     | LC     | FF     | PSEL3  | PSEL2 | PSEL1 | PSEL0 |

Register Array Bit Map Addresses 11H to 1FH Page (0)

| Address (Hex) | D7     | D6    | D5    | D4     | D3     | D2      | D1     | D0     |
|---------------|--------|-------|-------|--------|--------|---------|--------|--------|
| 11            | 1      | 0     | 0     | 0      | 0      | 0       | X      | X      |
| 12            | BDLNKC | PARTC | RECC  | SEC    | NSFDC  | PLERC   | ELBERC | JABC   |
| 13            | resv   | resv  | OWCC  | RXCOLC | TXCOLC | resv    | FWF    | ROR    |
| 14            | BDLNKE | PARTE | OWCE  | SEE    | NSFDE  | PLERE   | ELBERE | JABE   |
| 15            | res    | ISAM  | FWF-2 | ROR-2  | OWCC-2 | PARTC-2 | FAEC   | FCSC   |
| 16            | IFC    | IHC   | ILC   | IFF    | IREC   | ICOL    | IPART  | MIFCON |
| 17            | A5     | A4    | A3    | A2     | A1     | A0      | resv   | resv   |
| 18            | PCD7   | PCD6  | PCD5  | PCD4   | PCD3   | PCD2    | PCD1   | PCD0   |
| 1D            | res    | GLME  | res   | DSM    | res    | MPS     | GRP    | SAC    |
| 1F            | IFGT7  | IFGT6 | IFGT5 | IFGT4  | IFGT3  | IFGT2   | IFGT1  | IFGT0  |

Register Array Bit Map Addresses 11H to 1FH Page (1)

| Address (Hex) | D7    | D6   | D5  | D4   | D3   | D2   | D1    | D0  |
|---------------|-------|------|-----|------|------|------|-------|-----|
| 11 to 1D      | BDLNK | PART | OWC | SE   | NSFD | PLER | ELBER | JAB |
| 1E            | ER8   | ER7  | ER6 | ER5  | ER4  | ER3  | ER2   | ER1 |
| 1F            | DLU   | res  | res | ER13 | ER12 | ER11 | ER10  | ER9 |

Register Array Bit Map Addresses 11H to 1FH Pages (2) and (3)

| Address (Hex)  | D7   | D6   | D5   | D4   | D3   | D2   | D1  | D0  |
|----------------|------|------|------|------|------|------|-----|-----|
| 11             | —    | —    | —    | —    | —    | —    | —   | —   |
| Even Locations | EC7  | EC6  | EC5  | EC4  | EC3  | EC2  | EC1 | EC0 |
| Odd Locations  | EC15 | EC14 | EC13 | EC12 | EC11 | EC10 | EC9 | EC8 |



## 8.0 RIC II Registers (Continued)

### Register Array Bit Map Addresses 11H to 1FH Page (4)

| Address (Hex) | D7  | D6   | D5   | D4   | D3  | D2   | D1   | D0   |
|---------------|-----|------|------|------|-----|------|------|------|
| 11 to 1D      | EC7 | EC6  | EC5  | EC4  | EC3 | EC2  | EC1  | EC0  |
| 1E            | res | EDA  | ESA  | SAM  | MCE | BCE  | SME  | LME  |
| 1F            | ADV | PTR2 | PTR1 | PTR0 | ADV | PTR2 | PTR1 | PTR0 |

### Register Array Bit Map Addresses 11H to 1FH Page (5)

| Address (Hex) | D7           | D6           | D5           | D4           | D3           | D2           | D1           | D0           |
|---------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| 11, 12        | PCAMx<br>_D7 | PCAMx<br>_D6 | PCAMx<br>_D5 | PCAMx<br>_D4 | PCAMx<br>_D3 | PCAMx<br>_D2 | PCAMx<br>_D1 | PCAMx<br>_D0 |
| 13            | res          | EDA          | ESA          | SAM          | MCE          | BCE          | SME          | LME          |
| 14            | ADV          | PTR2         | PTR1         | PTR0         | ADV          | PTR2         | PTR1         | PTR0         |
| 15, 16        | PCAMx<br>_D7 | PCAMx<br>_D6 | PCAMx<br>_D5 | PCAMx<br>_D4 | PCAMx<br>_D3 | PCAMx<br>_D2 | PCAMx<br>_D1 | PCAMx<br>_D0 |
| 17            | res          | EDA          | ESA          | SAM          | MCE          | BCE          | SME          | LME          |
| 18            | ADV          | PTR2         | PTR1         | PTR0         | ADV          | PTR2         | PTR1         | PTR0         |
| 19, 1A        | PCAMx<br>_D7 | PCAMx<br>_D6 | PCAMx<br>_D5 | PCAMx<br>_D4 | PCAMx<br>_D3 | PCAMx<br>_D2 | PCAMx<br>_D1 | PCAMx<br>_D0 |
| 1B            | res          | EDA          | ESA          | SAM          | MCE          | BCE          | SME          | LME          |
| 1C            | ADV          | PTR2         | PTR1         | PTR0         | ADV          | PTR2         | PTR1         | PTR0         |
| 1D, 1E        | PCAMx<br>_D7 | PCAMx<br>_D6 | PCAMx<br>_D5 | PCAMx<br>_D4 | PCAMx<br>_D3 | PCAMx<br>_D2 | PCAMx<br>_D1 | PCAMx<br>_D0 |
| 1F            | res          | EDA          | ESA          | SAM          | MCE          | BCE          | SME          | LME          |

### Register Array Bit Map Addresses 11H to 1FH Page (6)

| Address (Hex) | D7           | D6           | D5           | D4           | D3           | D2           | D1           | D0           |
|---------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| 11            | ADV          | PTR2         | PTR1         | PTR0         | ADV          | PTR2         | PTR1         | PTR0         |
| 12, 13        | PCAMx<br>_D7 | PCAMx<br>_D6 | PCAMx<br>_D5 | PCAMx<br>_D4 | PCAMx<br>_D3 | PCAMx<br>_D2 | PCAMx<br>_D1 | PCAMx<br>_D0 |
| 14            | res          | EDA          | ESA          | SAM          | MCE          | BCE          | SME          | LME          |
| 15            | ADV          | PTR2         | PTR1         | PTR0         | ADV          | PTR2         | PTR1         | PTR0         |
| 16, 17        | PCAMx<br>_D7 | PCAMx<br>_D6 | PCAMx<br>_D5 | PCAMx<br>_D4 | PCAMx<br>_D3 | PCAMx<br>_D2 | PCAMx<br>_D1 | PCAMx<br>_D0 |
| 18            | res          | EDA          | ESA          | SAM          | MCE          | BCE          | SME          | LME          |
| 19            | ADV          | PTR2         | PTR1         | PTR0         | ADV          | PTR2         | PTR1         | PTR0         |
| 1A, 1B        | PCAMx<br>_D7 | PCAMx<br>_D6 | PCAMx<br>_D5 | PCAMx<br>_D4 | PCAMx<br>_D3 | PCAMx<br>_D2 | PCAMx<br>_D1 | PCAMx<br>_D0 |
| 1C            | res          | EDA          | ESA          | SAM          | MCE          | BCE          | SME          | LME          |
| 1D            | ADV          | PTR2         | PTR1         | PTR0         | ADV          | PTR2         | PTR1         | PTR0         |
| 1E, 1F        | PCAMx<br>_D7 | PCAMx<br>_D6 | PCAMx<br>_D5 | PCAMx<br>_D4 | PCAMx<br>_D3 | PCAMx<br>_D2 | PCAMx<br>_D1 | PCAMx<br>_D0 |

## 8.0 RIC II Registers (Continued)

### Register Array Bit Map Addresses 11H to 1FH Page (8)

| Address (Hex) | D7           | D6           | D5           | D4           | D3           | D2           | D1           | D0           |
|---------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| 11            | res          | EDA          | ESA          | SAM          | MCE          | BCE          | SME          | LME          |
| 12            | ADV          | PTR2         | PTR1         | PTR0         | ADV          | PTR2         | PTR1         | PTR0         |
| 13, 14        | PCAMx<br>_D7 | PCAMx<br>_D6 | PCAMx<br>_D5 | PCAMx<br>_D4 | PCAMx<br>_D3 | PCAMx<br>_D2 | PCAMx<br>_D1 | PCAMx<br>_D0 |
| 15            | res          | EDA          | ESA          | SAM          | MCE          | BCE          | SME          | LME          |
| 16            | ADV          | PTR2         | PTR1         | PTR0         | ADV          | PTR2         | PTR1         | PTR0         |
| 17, 18        | PCAMx<br>_D7 | PCAMx<br>_D6 | PCAMx<br>_D5 | PCAMx<br>_D4 | PCAMx<br>_D3 | PCAMx<br>_D2 | PCAMx<br>_D1 | PCAMx<br>_D0 |
| 19            | res          | EDA          | ESA          | SAM          | MCE          | BCE          | SME          | LME          |
| 1A            | ADV          | PTR2         | PTR1         | PTR0         | ADV          | PTR2         | PTR1         | PTR0         |
| 1B, 1C        | PCAMx<br>_D7 | PCAMx<br>_D6 | PCAMx<br>_D5 | PCAMx<br>_D4 | PCAMx<br>_D3 | PCAMx<br>_D2 | PCAMx<br>_D1 | PCAMx<br>_D0 |
| 1D            | res          | EDA          | ESA          | SAM          | MCE          | BCE          | SME          | LME          |
| 1E            | ADV          | PTR2         | PTR1         | PTR0         | ADV          | PTR2         | PTR1         | PTR0         |
| 1F            | PCAMx<br>_D7 | PCAMx<br>_D6 | PCAMx<br>_D5 | PCAMx<br>_D4 | PCAMx<br>_D3 | PCAMx<br>_D2 | PCAMx<br>_D1 | PCAMx<br>_D0 |

Note: For Port CAM register bits (PCAMx\_D[7:0]) and Shared CAM register bits (SCAMx\_D[7:0]) x represents the port number.

### Register Array Bit Map Addresses 11H to 1FH Page (9)

| Address (Hex) | D7           | D6           | D5           | D4           | D3           | D2           | D1           | D0           |
|---------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| 11            | PCAMx<br>_D7 | PCAMx<br>_D6 | PCAMx<br>_D5 | PCAMx<br>_D4 | PCAMx<br>_D3 | PCAMx<br>_D2 | PCAMx<br>_D1 | PCAMx<br>_D0 |
| 12            | res          | EDA          | ESA          | SAM          | MCE          | BCE          | SME          | LME          |
| 13            | ADV          | PTR2         | PTR1         | PTR0         | ADV          | PTR2         | PTR1         | PTR0         |
| 14, 15        | PCAMx<br>_D7 | PCAMx<br>_D6 | PCAMx<br>_D5 | PCAMx<br>_D4 | PCAMx<br>_D3 | PCAMx<br>_D2 | PCAMx<br>_D1 | PCAMx<br>_D0 |
| 16            | ADV8         | ADV7         | ADV6         | ADV5         | ADV4         | ADV3         | ADV2         | ADV1         |
| 17            | ADV16        | ADV15        | ADV14        | ADV13        | ADV12        | ADV11        | ADV10        | ADV9         |
| 18            | ADV24        | ADV23        | ADV22        | ADV21        | ADV20        | ADV19        | ADV18        | ADV17        |
| 19            | ADV32        | ADV31        | ADV30        | ADV29        | ADV28        | ADV27        | ADV26        | ADV25        |
| 1A            | SCAMx<br>_D7 | SCAMx<br>_D6 | SCAMx<br>_D5 | SCAMx<br>_D4 | SCAMx<br>_D3 | SCAMx<br>_D2 | SCAMx<br>_D1 | SCAMx<br>_D0 |
| 1B            | P8           | P7           | P6           | P5           | P4           | P3           | P2           | P1           |
| 1C            | PTR2         | PTR1         | PTR0         | P13          | P12          | P11          | P10          | P9           |
| 1D            | SCAMx<br>_D7 | SCAMx<br>_D6 | SCAMx<br>_D5 | SCAMx<br>_D4 | SCAMx<br>_D3 | SCAMx<br>_D2 | SCAMx<br>_D1 | SCAMx<br>_D0 |
| 1E            | P8           | P7           | P6           | P5           | P4           | P3           | P2           | P1           |
| 1F            | PTR2         | PTR1         | PTR0         | P13          | P12          | P11          | P10          | P9           |

## 8.0 RIC II Registers (Continued)

### Register Array Bit Map Addresses 11H to 1FH Pages (AH, BH, CH, DH, EH, FH)

| Address (Hex) | D7           | D6           | D5           | D4           | D3           | D2           | D1           | D0           |
|---------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| 11            | SCAMx<br>_D7 | SCAMx<br>_D6 | SCAMx<br>_D5 | SCAMx<br>_D4 | SCAMx<br>_D3 | SCAMx<br>_D2 | SCAMx<br>_D1 | SCAMx<br>_D0 |
| 12            | P8           | P7           | P6           | P5           | P4           | P3           | P2           | P1           |
| 13            | PTR2         | PTR1         | PTR0         | P13          | P12          | P11          | P10          | P9           |
| 14            | SCAMx<br>_D7 | SCAMx<br>_D6 | SCAMx<br>_D5 | SCAMx<br>_D4 | SCAMx<br>_D3 | SCAMx<br>_D2 | SCAMx<br>_D1 | SCAMx<br>_D0 |
| 15            | P8           | P7           | P6           | P5           | P4           | P3           | P2           | P1           |
| 16            | PTR2         | PTR1         | PTR0         | P13          | P12          | P11          | P10          | P9           |
| 17            | SCAMx<br>_D7 | SCAMx<br>_D6 | SCAMx<br>_D5 | SCAMx<br>_D4 | SCAMx<br>_D3 | SCAMx<br>_D2 | SCAMx<br>_D1 | SCAMx<br>_D0 |
| 18            | P8           | P7           | P6           | P5           | P4           | P3           | P2           | P1           |
| 19            | PTR2         | PTR1         | PTR0         | P13          | P12          | P11          | P10          | P9           |
| 1A            | SCAMx<br>_D7 | SCAMx<br>_D6 | SCAMx<br>_D5 | SCAMx<br>_D4 | SCAMx<br>_D3 | SCAMx<br>_D2 | SCAMx<br>_D1 | SCAMx<br>_D0 |
| 1B            | P8           | P7           | P6           | P5           | P4           | P3           | P2           | P1           |
| 1C            | PTR2         | PTR1         | PTR0         | P13          | P12          | P11          | P10          | P9           |
| 1D            | SCAMx<br>_D7 | SCAMx<br>_D6 | SCAMx<br>_D5 | SCAMx<br>_D4 | SCAMx<br>_D3 | SCAMx<br>_D2 | SCAMx<br>_D1 | SCAMx<br>_D0 |
| 1E            | P8           | P7           | P6           | P5           | P4           | P3           | P2           | P1           |
| 1F            | PTR2         | PTR1         | PTR0         | P13          | P12          | P11          | P10          | P9           |

Note: For Port CAM register bits (PCAMx\_D[7:0]) and Shared CAM register bits (SCAMx\_D[7:0]) x represents the port number.

## 8.0 RIC II Registers (Continued)

### RIC II STATUS AND CONFIGURATION REGISTER (ADDRESS 00H)

The lower portion of this register contains real time information concerning the operation of the RIC II. The upper three bits represent the chosen configuration of the transceiver interface employed.

| D7   | D6     | D5     | D4                        | D3                      | D2                       | D1                       | D0   |
|------|--------|--------|---------------------------|-------------------------|--------------------------|--------------------------|------|
| BINV | BYPAS2 | BYPAS1 | $\overline{\text{APART}}$ | $\overline{\text{JAB}}$ | $\overline{\text{AREC}}$ | $\overline{\text{ACOL}}$ | resv |

| Bit      | R/W    | Symbol                    | Description                                                                                                                                                                                                        |
|----------|--------|---------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0       | R      | resv                      | <b>RESERVED FOR FUTURE USE:</b> Reads as a logic 1.                                                                                                                                                                |
| D1       | R      | $\overline{\text{ACOL}}$  | <b>ANY COLLISIONS:</b><br>0: A collision is occurring at one or more of the RIC II's ports.<br>1: No collisions.                                                                                                   |
| D2       | R      | $\overline{\text{AREC}}$  | <b>ANY RECEIVE:</b><br>0: One of the RIC II's ports is the current packet or collision receiver.<br>1: No packet or collision reception within this RIC II.                                                        |
| D3       | R      | $\overline{\text{JAB}}$   | <b>JABBER PROTECT:</b><br>0: The RIC II has been forced into jabber protect state by one of its ports or by another port on the Inter-RIC bus (Multi-RIC II operations).<br>1: No jabber protect conditions exist. |
| D4       | R      | $\overline{\text{APART}}$ | <b>ANY PARTITION:</b><br>0: One or more ports are partitioned.<br>1: No ports are partitioned.                                                                                                                     |
| D5<br>D6 | R<br>R | BYPAS1<br>BYPAS2          | These bits define the configuration of ports 2 to 13, i.e., their use if the internal 10BASE-T transceivers or the external (AUI-like) transceiver interface.                                                      |
| D7       | R      | BINV                      | <b>BUS INVERT:</b> This register bit informs whether the Inter-RIC signals: IRE, ACTN, ANYXN, COLN and Management bus signal MCRS are:<br>0: Active high.<br>1: Active low.                                        |

## 8.0 RIC II Registers (Continued)

### POR1 REAL TIME STATUS REGISTERS (ADDRESS 01H TO 0DH)

|       |     |        |        |      |     |     |       |
|-------|-----|--------|--------|------|-----|-----|-------|
| D7    | D6  | D5     | D4     | D3   | D2  | D1  | D0    |
| DISPT | SQL | PTYPE1 | PTYPE0 | PART | REC | COL | GDLNK |

| Bit    | R/W    | Symbol                                                             | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |        |        |             |   |   |                                      |   |   |                                              |   |   |                                                                    |   |   |                                |
|--------|--------|--------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|--------|-------------|---|---|--------------------------------------|---|---|----------------------------------------------|---|---|--------------------------------------------------------------------|---|---|--------------------------------|
| D0     | R/W    | $\overline{\text{GDLNK}}$                                          | <b>GOOD LINK:</b><br>0: Link pulses are being received by the port.<br>1: Link pulses are not being received by the port logic. (Note 1)                                                                                                                                                                                                                                                                                                                                                  |        |        |             |   |   |                                      |   |   |                                              |   |   |                                                                    |   |   |                                |
| D1     | R      | $\overline{\text{COL}}$                                            | <b>COLLISION:</b><br>0: A collision is happening or has occurred during the current packet.<br>1: No collisions have occurred as yet during this packet.                                                                                                                                                                                                                                                                                                                                  |        |        |             |   |   |                                      |   |   |                                              |   |   |                                                                    |   |   |                                |
| D2     | R      | $\overline{\text{REC}}$                                            | <b>RECEIVE:</b><br>0: This port is now or has been the receive source of packet or collision information for the current packet.<br>1: This port has not been the receive source during the current packet.                                                                                                                                                                                                                                                                               |        |        |             |   |   |                                      |   |   |                                              |   |   |                                                                    |   |   |                                |
| D3     | R/W    | $\overline{\text{PART}}$                                           | <b>PARTITION:</b><br>Writing a logic one to this bit forces segment reconnection and partition state machine reset. Writing a zero to this bit has no effect.<br>0: This port is partitioned.<br>1: This port is not partitioned.                                                                                                                                                                                                                                                         |        |        |             |   |   |                                      |   |   |                                              |   |   |                                                                    |   |   |                                |
| D(5,4) | R      | PTYPE0<br>PTYPE1                                                   | <b>PARTITION TYPE 0</b><br><b>PARTITION TYPE 1</b><br>The partition type bits provide information specifying why the port is partitioned.                                                                                                                                                                                                                                                                                                                                                 |        |        |             |   |   |                                      |   |   |                                              |   |   |                                                                    |   |   |                                |
|        |        |                                                                    | <table border="1"> <thead> <tr> <th>PTYPE1</th> <th>PTYPE0</th> <th>Information</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Consecutive collision limit reached.</td> </tr> <tr> <td>0</td> <td>1</td> <td>Excessive length of collision limit reached.</td> </tr> <tr> <td>1</td> <td>0</td> <td>Failure to see data loopback from transceiver in monitored window.</td> </tr> <tr> <td>1</td> <td>1</td> <td>Processor forced reconnection.</td> </tr> </tbody> </table> | PTYPE1 | PTYPE0 | Information | 0 | 0 | Consecutive collision limit reached. | 0 | 1 | Excessive length of collision limit reached. | 1 | 0 | Failure to see data loopback from transceiver in monitored window. | 1 | 1 | Processor forced reconnection. |
| PTYPE1 | PTYPE0 | Information                                                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |        |        |             |   |   |                                      |   |   |                                              |   |   |                                                                    |   |   |                                |
| 0      | 0      | Consecutive collision limit reached.                               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |        |        |             |   |   |                                      |   |   |                                              |   |   |                                                                    |   |   |                                |
| 0      | 1      | Excessive length of collision limit reached.                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |        |        |             |   |   |                                      |   |   |                                              |   |   |                                                                    |   |   |                                |
| 1      | 0      | Failure to see data loopback from transceiver in monitored window. |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |        |        |             |   |   |                                      |   |   |                                              |   |   |                                                                    |   |   |                                |
| 1      | 1      | Processor forced reconnection.                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |        |        |             |   |   |                                      |   |   |                                              |   |   |                                                                    |   |   |                                |
| D6     | R/W    | SQL                                                                | <b>SQUELCH LEVELS:</b> (Notes 2 and 3)<br>0: Port operates with normal IEEE receive squelch level.<br>1: Port operates with reduced receive squelch level.                                                                                                                                                                                                                                                                                                                                |        |        |             |   |   |                                      |   |   |                                              |   |   |                                                                    |   |   |                                |
| D7     | R/W    | DISPT                                                              | <b>DISABLE PORT:</b><br>0: Port operates as defined by repeater operations.<br>1: All port activity is prevented.                                                                                                                                                                                                                                                                                                                                                                         |        |        |             |   |   |                                      |   |   |                                              |   |   |                                                                    |   |   |                                |

**Note 1:** Writing a 1 to this bit will cause the 10Base-T transceiver not to transmit or monitor the reception of link pulses. If the internal 10BASE-T transceivers are not selected or if port 1 (AUI port) is read, then this bit is undefined.

**Note 2:** This bit has no effect when external transceiver is selected.

**Note 3 (for RIC II only):** In addition to hysteresis that DP83950 RIC provides on normal receive squelch, DP83952 RIC II provides a hysteresis when operating in the reduced squelch level mode.

## 8.0 RIC II Registers (Continued)

### RIC II CONFIGURATION REGISTER (ADDRESS 0EH)

This register displays the state of a number of RIC II configuration bits loaded during the Mode Load operation.

| D7     | D6    | D5     | D4   | D3     | D2    | D1  | D0  |
|--------|-------|--------|------|--------|-------|-----|-----|
| MINMAX | DPART | TXONLY | OWCE | LPPART | CCLIM | TW2 | GSE |

| Bit | R/W | Symbol | Description                                                                                                                                                                                                                                                                                                                                                                              |
|-----|-----|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0  | R   | GSE    | <b>GLOBAL SECURITY ENABLE:</b><br>0: RIC II operates in security mode with Learning Mode enabled by default for all ports.<br>1: RIC II operates in non-security mode.                                                                                                                                                                                                                   |
| D1  | R   | TW2    | <b>CARRIER RECOVERY TIME:</b><br>0: TW2 set at 5 bits.<br>1: TW2 set at 3 bits.                                                                                                                                                                                                                                                                                                          |
| D2  | R   | CCLIM  | <b>CONSECUTIVE COLLISION LIMIT:</b><br>0: Consecutive collision limit set at 63 collisions.<br>1: Consecutive collision limit set at 31 collisions.                                                                                                                                                                                                                                      |
| D3  | R   | LPPART | <b>LOOPBACK PARTITION:</b><br>0: Partitioning upon lack of loopback from transceivers is enabled.<br>1: Partitioning upon lack of loopback from transceivers is disabled.                                                                                                                                                                                                                |
| D4  | R   | OWCE   | <b>OUT OF WINDOW COLLISION ENABLE:</b><br>0: Out of window collisions are treated as in window collisions by the segment partition state machines.<br>1: Out of window collisions are treated as out of window collisions by the segment partition state machines.                                                                                                                       |
| D5  | R   | TXONLY | <b>ONLY RECONNECT UPON SEGMENT TRANSMISSION:</b><br>0: A segment will only be reconnected to the network if a packet transmitted by the RIC II onto that segment fulfills the requirements of the segment reconnection algorithm.<br>1: A segment will be reconnected to the network by any packet on the network which fulfills the requirements of the segment reconnection algorithm. |
| D6  | R   | DPART  | <b>DISABLE PARTITION:</b><br>0: Partitioning of ports by on-chip algorithms is prevented.<br>1: Partitioning of ports by on-chip algorithms is enabled.                                                                                                                                                                                                                                  |
| D7  | R   | MINMAX | <b>MINIMUM/MAXIMUM DISPLAY MODE:</b><br>0: LED display set in minimum display mode.<br>1: LED display set in maximum display mode.                                                                                                                                                                                                                                                       |

## 8.0 RIC II Registers (Continued)

### REAL TIME INTERRUPT REGISTER (ADDRESS 0FH)

The Real Time Interrupt register (RTI) contains information which may change on a packet by packet basis. Any remaining interrupts which have not been serviced before the following packet is transmitted are cleared. Since multiple interrupt sources may be displayed by the RTI a priority scheme is implemented. A read cycle to the RTI gives the interrupt source and an address vector indicating the RIC II port which generated the interrupt.

The order of priority for the display of interrupt information is as follows (in secure mode only):

1. **Source Address Mismatch (added feature to the RIC II which is not present in the RIC),**
2. The receive source of network activity (Port N),
3. The first RIC II port showing collision,
4. A port partitioned or reconnected.

During the repetition of a single packet it is possible that multiple ports may be partitioned or alternatively reconnected. The ports have equal priority in displaying partition/reconnection information. This data is derived from the ports by the RTI register as it polls consecutively around the ports.

Reading the RTI clears the particular interrupt. If no interrupt sources are active the RTI returns a no valid interrupt status.

|        |        |        |        |       |       |       |       |
|--------|--------|--------|--------|-------|-------|-------|-------|
| D7     | D6     | D5     | D4     | D3    | D2    | D1    | D0    |
| IVCTR3 | IVCTR2 | IVCTR1 | IVCTR0 | ISRC3 | ISRC2 | ISRC1 | ISRC0 |

| Bit    | R/W | Symbol     | Description                                                                                            |
|--------|-----|------------|--------------------------------------------------------------------------------------------------------|
| D(3:0) | R   | ISCR(3:0)  | <b>INTERRUPT SOURCE:</b> These four bits indicate the reason why the interrupt was generated.          |
| D(7:4) | R   | IVCTR(3:0) | <b>INTERRUPT VECTOR:</b> This field defines the port address responsible for generating the interrupt. |

The following table shows the mapping of interrupt sources onto the D3 to D0 pins. Essentially each of the three interrupt sources has a dedicated bit in this field. If a read to the RTI produces a low logic level on one of these bits then the interrupt source may be directly decoded. Associated with the source of the interrupt is the port where the event is occurring. If no unmasked events (receive, collision, etc.), have occurred when the RTI is read then an all ones pattern is driven by the RIC onto the data pins.

| D7  | D6  | D5  | D4  | D3 | D2 | D1 | D0 | Comments                                                                  |
|-----|-----|-----|-----|----|----|----|----|---------------------------------------------------------------------------|
| PA3 | PA2 | PA1 | PA0 | 1  | 1  | 0  | 1  | <b>Source Address Mismatch</b><br>PA(3:0) = Port Address for the Mismatch |
| PA3 | PA2 | PA1 | PA0 | 1  | 1  | 0  | 1  | First Collision<br>PA(3:0) = Collision Port Address                       |
| PA3 | PA2 | PA1 | PA0 | 1  | 0  | 1  | 1  | Receive<br>PA(3:0) = Receive Port Address                                 |
| PA3 | PA2 | PA1 | PA0 | 0  | 1  | 1  | 1  | Partition Reconnection<br>PA(3:0) = Partition Port Address                |
| 1   | 1   | 1   | 1   | 1  | 1  | 1  | 1  | No Valid Interrupt                                                        |

## 8.0 RIC II Registers (Continued)

### PAGE SELECT REGISTER ((ALL PAGES) ADDRESS 10H)

The Page Select register performs two functions:

1. It enables switches to be made between register pages,
2. It provides status information regarding the Event Logging Interrupts.

| D7 | D6 | D5 | D4 | D3    | D2    | D1    | D0    |
|----|----|----|----|-------|-------|-------|-------|
| FC | HC | LC | FF | PSEL3 | PSEL2 | PSEL1 | PSEL0 |

| Bit    | R/W | Symbol    | Description                                                                                                                                                         |
|--------|-----|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D(3:0) | R/W | PSEL(3:0) | <b>PAGE SELECT BITS:</b> When read these bits indicate the currently selected Upper Register Array Page. Write cycles to these locations facilitates page swapping. |
| D4     | R   | FF        | <b>FLAG FOUND:</b> This indicates one of the unmasked event recording latches has been set.                                                                         |
| D5     | R   | LC        | <b>LOW COUNT:</b> This indicates one of the port event counters has a value less than 00FF Hex.                                                                     |
| D6     | R   | HC        | <b>HIGH COUNT:</b> This indicates one of the port event counters has a value greater than C000 Hex.                                                                 |
| D7     | R   | FC        | <b>FULL COUNTER:</b> This indicates one of the port event counters has a value equal to FFFF Hex.                                                                   |

### DEVICE TYPE REGISTER (PAGE 0H ADDRESS 11H)

This register may be used to distinguish different revisions of RIC. It will return the value D7 = 1 for the DP83952 RIC II device, or the value D7 = 0 for the DP83950 RIC device. Write operations to this register have no effect upon the contents.

| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
|----|----|----|----|----|----|----|----|
| 1  | 0  | 0  | 0  | 0  | 0  | X  | X  |



## 8.0 RIC II Registers (Continued)

### LOWER EVENT COUNT MASK REGISTER (PAGE 0H ADDRESS 12H)

|        |       |      |     |       |       |        |      |
|--------|-------|------|-----|-------|-------|--------|------|
| D7     | D6    | D5   | D4  | D3    | D2    | D1     | D0   |
| BDLNKC | PARTC | RECC | SEC | NSFDC | PLERC | ELBERC | JABC |

| Bit | R/W | Symbol | Description                                                                                                            |
|-----|-----|--------|------------------------------------------------------------------------------------------------------------------------|
| D0  | R/W | JABC   | <b>JABBER COUNT ENABLE:</b> Enables recording of Jabber Protect events.                                                |
| D1  | R/W | ELBERC | <b>ELASTICITY BUFFER ERROR COUNT ENABLE:</b> Enables recording of Elasticity Buffer Error events.                      |
| D2  | R/W | PLERC  | <b>PHASE LOCK ERROR COUNT ENABLE:</b> Enables recording of Carrier Error events.                                       |
| D3  | R/W | NSFDC  | <b>NON-SFD COUNT ENABLE:</b> Enables recording of Non-SFD packet events.                                               |
| D4  | R/W | SEC    | <b>SHORT EVENT COUNT ENABLE:</b> Enables recording of Short events.                                                    |
| D5  | R/W | RECC   | <b>RECEIVE COUNT ENABLE:</b> Enables recording of Packet Receive (Port N status) events that do not suffer collisions. |
| D6  | R/W | PARTC  | <b>PARTITION COUNT ENABLE:</b> Enables recording of Partition events.                                                  |
| D7  | R/W | BDLNKC | <b>BAD LINK COUNT ENABLE:</b> Enables recording of Bad Link events.                                                    |

### UPPER EVENT COUNT MASK REGISTER (PAGE 0H ADDRESS 13H)

The bits in this register effect the Upper and Lower Port Event Count Registers (ECR) on Page (3) addresses 12H to 1FH, and Page (4) addresses 12H to 1DH.

|      |      |      |        |        |      |     |     |
|------|------|------|--------|--------|------|-----|-----|
| D7   | D6   | D5   | D4     | D3     | D2   | D1  | D0  |
| resv | resv | OWCC | RXCOLC | TXCOLC | resv | FWF | ROR |

| Bit     | R/W | Symbol | Description                                                                                                                                                                      |
|---------|-----|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0      | R/W | ROR    | <b>RESET ON READ:</b> This bit selects the action a read operation has upon a port's event counter:<br>0: No effect upon register contents.<br>1: The counter register is reset. |
| D1      | R/W | FWF    | <b>FREEZE WHEN FULL:</b> This bit controls the freezing of the Event Count registers when the counter is full (FFFF Hex).                                                        |
| D2      | R   | resv   | <b>RESERVED FOR FUTURE USE:</b> This bit should be written with a low logic level.                                                                                               |
| D3      | R/W | TXCOLC | <b>TRANSMIT COLLISION COUNT ENABLE:</b> Enables recording of transmit collision events only.                                                                                     |
| D4      | R/W | RXCOLC | <b>RECEIVE COLLISION COUNT ENABLE:</b> Enables recording of receive collision events only.                                                                                       |
| D5      | R/W | OWCC   | <b>OUT OF WINDOW COLLISION COUNT ENABLE:</b> Enables recording of out of window collision events only.                                                                           |
| D(7: 6) | R   | resv   | <b>RESERVED FOR FUTURE USE:</b> These bits should be written with a low logic level.                                                                                             |

**Note:** To count all collisions then both the TXCOLC and RXCOLC bits must be set. The OWCC bit should not be set otherwise the port counter will be incremented twice when an out of collision window collision occurs. The OWCC bit alone should be set if only out of window collision are to be counted.

## 8.0 RIC II Registers (Continued)

### EVENT RECORD MASK REGISTER (PAGE 0H ADDRESS 14H)

| D7     | D6    | D5   | D4  | D3    | D2    | D1     | D0   |
|--------|-------|------|-----|-------|-------|--------|------|
| BDLNKE | PARTE | OWCE | SEE | NSFDE | PLERE | ELBERE | JABE |

| Bit | R/W | Symbol | Description                                                                                            |
|-----|-----|--------|--------------------------------------------------------------------------------------------------------|
| D0  | R/W | JABE   | <b>JABBER ENABLE:</b> Enables recording of Jabber Protect events.                                      |
| D1  | R/W | ELBERE | <b>ELASTICITY BUFFER ERROR ENABLE:</b> Enables recording of Elasticity Buffer Error events.            |
| D2  | R/W | PLERE  | <b>PHASE LOCK ERROR ENABLE:</b> Enables recording of Carrier Error events.                             |
| D3  | R/W | NSFDE  | <b>NON-SFD ENABLE:</b> Enables recording of Non-SFD packet events.                                     |
| D4  | R/W | SEE    | <b>SHORT EVENT ENABLE:</b> Enables recording of Short Events.                                          |
| D5  | R/W | OWCE   | <b>OUT OF WINDOW COLLISION COUNT ENABLE:</b> Enables recording of Out of Window Collision events only. |
| D6  | R/W | PARTE  | <b>PARTITION ENABLE:</b> Enables recording of Partition events.                                        |
| D7  | R/W | BDLNKE | <b>BAD LINK ENABLE:</b> Enables recording of Bad Link Events.                                          |

Note: Writing a 1 enables the event to be recorded.

### EVENT COUNT AND INTERRUPT MASK REGISTER 2 (ECIMR-2) (PAGE 0H ADDRESS 15H)

The bits in this register effect the Port Event Count Register 2, PECCR-2 on Page 4, Addresses 11H to 1DH.

| D7  | D6   | D5    | D4    | D3     | D2      | D1   | D0   |
|-----|------|-------|-------|--------|---------|------|------|
| res | ISAM | FWF-2 | ROR-2 | OWCC-2 | PARTC-2 | FAEC | FCSC |

| Bit | R/W | Symbol  | Description                                                                                                                                                                                                                                  |
|-----|-----|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0  | R/W | FCSC    | <b>FRAME CHECK SEQUENCE COUNT ENABLE:</b> This bit enables counting the packets with frame check sequence error.<br>0: Disable the frame check sequence count.<br>1: Enable the frame check sequence count.                                  |
| D1  | R/W | FAEC    | <b>FRAME ALIGNMENT ERROR COUNT ENABLE:</b> This bit enables counting the packets with frame alignment error.<br>0: Disable the frame alignment error count.<br>1: Enable the frame alignment error count.                                    |
| D2  | R/W | PARTC-2 | <b>PARTITION COUNT ENABLE:</b> This bit enables recording of partition events.<br>0: Disable the partition count.<br>1: Enable the partition count.                                                                                          |
| D3  | R/W | OWCC-2  | <b>OUT OF WINDOW COLLISION COUNT ENABLE:</b> This bit enables counting of out of window collision events.<br>0: Disable the out of window collision count.<br>1: Enable the out of window collision count.                                   |
| D4  | R/W | ROR-2   | <b>RESET ON READ:</b> This bit enables the counter register to reset upon reading the port event's counter.<br>0: No effect upon reading the register contents.<br>1: The counter register is reset by reading the contents of the register. |
| D5  | R/W | FWF-2   | <b>FREEZE WHEN FULL:</b> This bit controls the freezing of the Event Count registers when the counter is full (FF Hex).<br>0: No effect on the event count register.<br>1: Freeze the event count register when the counter is full.         |
| D6  | R/W | ISAM    | <b>INTERRUPT ON THE SOURCE ADDRESS MISMATCH MASK:</b><br>0: Interrupts will be generated on a source address mismatch mask. ( $\overline{RTI}$ pin becomes active.)<br>1: No interrupts are generated.                                       |
| D7  | R   | res     | <b>RESERVED FOR FUTURE USE:</b> Reads as a logic 0.                                                                                                                                                                                          |

## 8.0 RIC II Registers (Continued)

### INTERRUPT AND MANAGEMENT CONFIGURATION REGISTER (PAGE 0H ADDRESS 16H)

This register powers up with all bits set to one and must be initialized by a processor write cycle before any events will generate interrupts.

|     |     |     |     |      |      |       |        |
|-----|-----|-----|-----|------|------|-------|--------|
| D7  | D6  | D5  | D4  | D3   | D2   | D1    | D0     |
| IFC | IHC | ILC | IFF | IREC | ICOL | IPART | MIFCON |

| Bit | R/W | Symbol | Description                                                                                                                                                                                                                                                                                                                                                                                            |
|-----|-----|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0  | R/W | MIFCON | <b>MANAGEMENT INTERFACE CONFIGURATION:</b><br>0: All Packets repeated are transmitted over the Management bus.<br>1: Packets repeated by the RIC II which do not have a Start of Frame Delimiters are not transmitted over the Management bus.                                                                                                                                                         |
| D1  | R/W | IPART  | <b>INTERRUPT ON PARTITION:</b><br>0: Interrupts will be generated ( $\overline{RTI}$ pin goes active) if a port becomes Partitioned.<br>1: No interrupts are generated by this condition.                                                                                                                                                                                                              |
| D2  | R/W | ICOL   | <b>INTERRUPT ON COLLISION:</b><br>0: Interrupts will be generated ( $\overline{RTI}$ pin goes active) if this RIC II has a port which experiences a collision, Single RIC II applications, or contains a port which experiences a receive collision or is the first port to suffer a transmit collision in a packet in Multi-RIC II applications.<br>1: No interrupts are generated by this condition. |
| D3  | R/W | IREC   | <b>INTERRUPT ON RECEIVE:</b><br>0: Interrupts will be generated ( $\overline{RTI}$ pin goes active) if this RIC II contains the receive port for packet or collision activity.<br>1: No interrupts are generated by this condition.                                                                                                                                                                    |
| D4  | R/W | IFF    | <b>INTERRUPT ON FLAG FOUND:</b><br>0: Interrupts will be generated ( $\overline{ELI}$ pin goes active) if one or more than one of the flags in the flag array is true.<br>1: No interrupts are generated by this condition.                                                                                                                                                                            |
| D5  | R/W | ILC    | <b>INTERRUPT ON LOW COUNT:</b><br>0: Interrupt generated ( $\overline{ELI}$ pin goes active) when one or more of the Event Counters holds a value less than 256 counts.<br>1: No effect                                                                                                                                                                                                                |
| D6  | R/W | IHC    | <b>INTERRUPT ON HIGH COUNT:</b><br>0: Interrupt generated ( $\overline{ELI}$ pin goes active) when one or more of the Event Counters holds a value in excess of 49,152 counts.<br>1: No effect.                                                                                                                                                                                                        |
| D7  | R/W | IFC    | <b>INTERRUPT ON FULL COUNTER:</b><br>0: Interrupt generated ( $\overline{ELI}$ pin goes active) when one or more of the Event Counters is full.<br>1: No effect.                                                                                                                                                                                                                                       |

## 8.0 RIC II Registers (Continued)

### RIC II ADDRESS REGISTER (PAGE 0H ADDRESS 17H)

This register may be used to differentiate between RIC IIs in a multi-RIC II repeater system. The contents of this register form part of the information available through the management bus.

|    |    |    |    |    |    |     |     |
|----|----|----|----|----|----|-----|-----|
| D7 | D6 | D5 | D4 | D3 | D2 | D1  | D0  |
| A5 | A4 | A3 | A2 | A1 | A0 | res | res |

### PACKET COMPRESS DECODE REGISTER (PAGE 0H ADDRESS 18H)

This register is used to determine the number of bytes in the data field of a packet which are transferred over the management bus when the packet compress option is employed. The register bits perform the function of a direct binary decode. Thus up to 255 bytes of data may be transferred over the management bus if packet compression is selected.

|      |      |      |      |      |      |      |      |
|------|------|------|------|------|------|------|------|
| D7   | D6   | D5   | D4   | D3   | D2   | D1   | D0   |
| PCD7 | PCD6 | PCD5 | PCD4 | PCD3 | PCD2 | PCD1 | PCD0 |

### GLOBAL SECURITY REGISTER (GSR) (PAGE 0H ADDRESS 1DH)

This register provides various security configuration options. For instance, enabling learning mode for all the ports; starting address comparison; using the modified packet status register 5 for the management bus; generating random pattern on source address mismatch; disabling port on source address mismatch.

|     |      |     |     |     |     |     |     |
|-----|------|-----|-----|-----|-----|-----|-----|
| D7  | D6   | D5  | D4  | D3  | D2  | D1  | D0  |
| res | GLME | res | DSM | res | MPS | GRP | SAC |

| Bit | R/W | Symbol | Description                                                                                                                                                                                                                                                         |
|-----|-----|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0  | R/W | SAC    | <b>START ADDRESS COMPARISON:</b><br>0: Do not begin comparison.<br>1: Begin comparison.                                                                                                                                                                             |
| D1  | R/W | GRP    | <b>GENERATE RANDOM PATTERN:</b> This bit controls generating the random pattern on a valid source address mismatch. In any event, the Hub Manager will be informed on the SA mismatch.<br>0: Generate the random pattern.<br>1: Do not generate the random pattern. |
| D2  | R/W | MPS    | <b>MODIFY PACKET STATUS REGISTER 5:</b> This bit enables modifying the packet status register 5, PSR5 on the 7 management bytes, over the management bus.<br>0: Do not modify the PSR5.<br>1: Modify the PSR5.                                                      |
| D3  | R   | res    | <b>RESERVED FOR FUTURE USE:</b> For proper operation, this bit must be 0.                                                                                                                                                                                           |
| D4  | R/W | DSM    | <b>DISABLE THE PORT ON A SOURCE ADDRESS MISMATCH:</b><br>0: Do not disable the port on a valid source address mismatch.<br>1: Disable the port on a valid source address mismatch.                                                                                  |
| D5  | R   | res    | <b>RESERVED FOR FUTURE USE</b>                                                                                                                                                                                                                                      |
| D6  | R/W | GLME   | <b>GLOBAL LEARN MODE ENABLE:</b> (Note)<br>0: Do not enable the learn mode for all ports.<br>1: Enable the learn mode for all ports.                                                                                                                                |
| D7  | R   | res    | <b>RESERVED FOR FUTURE USE:</b> Reads as a logic 0.                                                                                                                                                                                                                 |

**Note:** The GLME is not a status bit. Reading this bit indicates what value was last written to it.

## 8.0 RIC II Registers (Continued)

### INTER FRAME GAP THRESHOLD SELECT REGISTER (PAGE 0H ADDRESS 1FH)

This register is used to configure the hub management interface to provide a certain minimum inter frame gap between packets transmitted over the management bus. The value written to this register, plus one, is the magnitude in bit times of the minimum IFG allowed on the management bus.

|       |       |       |       |       |       |       |       |
|-------|-------|-------|-------|-------|-------|-------|-------|
| D7    | D6    | D5    | D4    | D3    | D2    | D1    | D0    |
| IFGT7 | IFGT6 | IFGT5 | IFGT4 | IFGT3 | IFGT2 | IFGT1 | IFGT0 |

### PORT EVENT RECORD REGISTERS (PAGE 1H ADDRESS 11H TO 1DH)

These registers hold the recorded events for the specified RIC II port. The flags are cleared when the register is read.

|       |      |     |    |      |      |       |     |
|-------|------|-----|----|------|------|-------|-----|
| D7    | D6   | D5  | D4 | D3   | D2   | D1    | D0  |
| BDLNK | PART | OWC | SE | NSFD | PLER | ELBER | JAB |

| Bit | R/W | Symbol | Description                                                                    |
|-----|-----|--------|--------------------------------------------------------------------------------|
| D0  | R   | JAB    | <b>JABBER:</b> A Jabber Protect event has occurred.                            |
| D1  | R   | ELBER  | <b>ELASTICITY BUFFER ERROR:</b> A Elasticity Buffer Error has occurred.        |
| D2  | R   | PLER   | <b>PHASE LOCK ERROR:</b> A Phase Lock Error event has occurred.                |
| D3  | R   | NSFD   | <b>NON-SFD:</b> A Non-SFD packet event has occurred.                           |
| D4  | R   | SE     | <b>SHORT EVENT:</b> A short event has occurred.                                |
| D5  | R   | OWC    | <b>OUT OF WINDOW COLLISION:</b> An out of window collision event has occurred. |
| D6  | R   | PART   | <b>PARTITION:</b> A partition event has occurred.                              |
| D7  | R   | BDLNK  | <b>BAD LINK:</b> A link failure event has occurred.                            |

## 8.0 RIC II Registers (Continued)

### UPPER EVENT INFORMATION REGISTER (UPPER EIR) (PAGE 1H ADDRESS 1EH)

| D7  | D6  | D5  | D4  | D3  | D2  | D1  | D0  |
|-----|-----|-----|-----|-----|-----|-----|-----|
| ER8 | ER7 | ER6 | ER5 | ER4 | ER3 | ER2 | ER1 |

| Bit | R/W | Symbol | Description                                                                                    |
|-----|-----|--------|------------------------------------------------------------------------------------------------|
| D0  | R   | ER1    | 0: Flag found not generated by event on port 1.<br>1: Flag found generated by event on port 1. |
| D1  | R   | ER2    | 0: Flag found not generated by event on port 2.<br>1: Flag found generated by event on port 2. |
| D2  | R   | ER3    | 0: Flag found not generated by event on port 3.<br>1: Flag found generated by event on port 3. |
| D3  | R   | ER4    | 0: Flag found not generated by event on port 4.<br>1: Flag found generated by event on port 4. |
| D4  | R   | ER5    | 0: Flag found not generated by event on port 5.<br>1: Flag found generated by event on port 5. |
| D5  | R   | ER6    | 0: Flag found not generated by event on port 6.<br>1: Flag found generated by event on port 6. |
| D6  | R   | ER7    | 0: Flag found not generated by event on port 7.<br>1: Flag found generated by event on port 7. |
| D7  | R   | ER8    | 0: Flag found not generated by event on port 8.<br>1: Flag found generated by event on port 8. |

### LOWER EVENT INFORMATION REGISTER (LOWER EIR) (PAGE 1H ADDRESS 1FH)

| D7  | D6  | D5  | D4   | D3   | D2   | D1   | D0  |
|-----|-----|-----|------|------|------|------|-----|
| DLU | res | res | ER13 | ER12 | ER11 | ER10 | ER9 |

| Bit | R/W | Symbol | Description                                                                                                                                                                                         |
|-----|-----|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0  | R   | ER9    | 0: Flag found not generated by event on port 9.<br>1: Flag found generated by event on port 9.                                                                                                      |
| D1  | R   | ER10   | 0: Flag found not generated by event on port 10.<br>1: Flag found generated by event on port 10.                                                                                                    |
| D2  | R   | ER11   | 0: Flag found not generated by event on port 11.<br>1: Flag found generated by event on port 11.                                                                                                    |
| D3  | R   | ER12   | 0: Flag found not generated by event on port 12.<br>1: Flag found generated by event on port 12.                                                                                                    |
| D4  | R   | ER13   | 0: Flag found not generated by event on port 13.<br>1: Flag found generated by event on port 13.                                                                                                    |
| D5  | R   | res    | <b>RESERVED FOR FUTURE USE:</b> Reads as a logic 0.                                                                                                                                                 |
| D6  | R   | res    | <b>RESERVED FOR FUTURE USE:</b> Reads as a logic 0.                                                                                                                                                 |
| D7  | R/W | DLU    | <b>DISABLE THE LED UPDATES:</b> This bit disables the LED display updates for a faster processor register access.<br>0: Re-enable the LED update cycle (Note).<br>1: Disable the LED update cycles. |

**Note:** The LED update cycle will be re-enabled only when the network and the RIC II internal state machines are idle.

## 8.0 RIC II Registers (Continued)

### PORT EVENT COUNT REGISTER (PAGES 2H AND 3H)

The Event Count (EC) register shows the instantaneous value of the specified port's 16-bit counter. The counter increments when an enabled event occurs. The counter may be cleared when it is read and prevented from rolling over when the maximum count is reached by setting the appropriate control bits in the Upper Event Count mask register. Since the RIC II's processor port is octal and the counters are 16 bits long a temporary holding register is employed for register reads. When one of the counters is read, either high or low byte first, all 16 bits of the counter are transferred to a holding register. Provided the next read cycle to the counter array accesses the same counter's other byte, then the read cycle accesses the holding register. This avoids the problem of events occurring in between the two processor reads and indicating a false count value. In order to enter a new value to the holding register a different counter must be accessed, or the same counter byte must be re-read.

#### Lower Byte

| D7  | D6  | D5  | D4  | D3  | D2  | D1  | D0  |
|-----|-----|-----|-----|-----|-----|-----|-----|
| EC7 | EC6 | EC5 | EC4 | EC3 | EC2 | EC1 | EC0 |

#### Upper Byte

| D7   | D6   | D5   | D4   | D3   | D2   | D1  | D0  |
|------|------|------|------|------|------|-----|-----|
| EC15 | EC14 | EC13 | EC12 | EC11 | EC10 | EC9 | EC8 |

### PORT EVENT COUNT REGISTER 2 (PECR-2) (PAGE 4H ADDRESSES 11H TO 1DH)

The Port Event Count Register 2 (PECR-2) shows the instantaneous value of the specified port's 8-bit counter. The counter increments when an enabled event occurs. The counter may be cleared when it is read, and prevented from rolling over when the maximum count is reached, by setting the appropriate control bits in the ECIMR-2 register.

| D7  | D6  | D5  | D4  | D3  | D2  | D1  | D0  |
|-----|-----|-----|-----|-----|-----|-----|-----|
| EC7 | EC6 | EC5 | EC4 | EC3 | EC2 | EC1 | EC0 |

## 8.0 RIC II Registers (Continued)

### PORT SECURITY CONFIGURATION REGISTER (PSCR) (PAGES 4H, 5H, 6H, 8H, 9H)

This register sets up the various security modes for the RIC II. It provides port specific information such as enabling/disabling the security mode, passing broadcast packets, etc. In addition, comparison on destination address, source address, or both can be selected. The system can also qualify learning mode on a per port basis.

**Note:** Bit D0 is only for the port CAMs, and not for the shared CAMs.

| D7  | D6  | D5  | D4  | D3  | D2  | D1  | D0  |
|-----|-----|-----|-----|-----|-----|-----|-----|
| res | EDA | ESA | SAM | MCE | BCE | SME | LME |

| Bit | R/W | Symbol | Description                                                                                                                                                                                                                                        |
|-----|-----|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0  | R/W | LME    | <b>LEARNING MODE:</b><br>0: Disable Learn Mode for port CAMs.<br>1: Enable Learn Mode for port CAMs.                                                                                                                                               |
| D1  | R/W | SME    | <b>SECURITY MODE:</b><br>0: Disable Security Mode.<br>1: Enable Security Mode.                                                                                                                                                                     |
| D2  | R/W | BCE    | <b>ACCEPT BROADCAST:</b> Enables the repeater to pass/repeat a packet with an all 1's destination address (Note 1).<br>0: Replace the broadcast packets with random packets.<br>1: Pass broadcast packets.                                         |
| D3  | R/W | MCE    | <b>ACCEPT MULTICAST:</b> Enables the repeater to pass/repeat a packet with the LSB of "1" in the most significant byte of the destination address (Note 2).<br>0: Replace the multicast packets with random packets.<br>1: Pass multicast packets. |
| D4  | R   | SAM    | <b>SOURCE ADDRESS MATCH/MISMATCH:</b><br>0: Source address match occurred for the packet.<br>1: Source address mismatch occurred for the packet.                                                                                                   |
| D5  | R/W | ESA    | <b>SOURCE ADDRESS SECURITY:</b><br>0: Do not employ source address to implement security.<br>1: Employ source address to implement security.                                                                                                       |
| D6  | R/W | EDA    | <b>DESTINATION ADDRESS SECURITY:</b><br>0: Do not employ destination address to implement security.<br>1: Employ destination address to implement security.                                                                                        |
| D7  | R   | res    | <b>RESERVED FOR FUTURE USE:</b> Reads as a logic 0.                                                                                                                                                                                                |

**Note 1:** SA mismatch is still valid for broadcast packets.

**Note 2:** SA mismatch is still valid for multicast packets.



## 8.0 RIC II Registers (Continued)

### PORT CAM POINTER REGISTER (PCPR) (PAGES 4H, 5H, 6H, 8H, 9H)

This register indicates which bytes of the six Ethernet address bytes has been stored in the CAM locations. When a byte has been loaded into the CAM location, the pointer increments. Thus, this register indicates which byte will get written on the subsequent CAM location access. After the complete address is stored in any of the two CAMs, the user must set the "address valid", ADV, bit so the address is not overwritten mistakenly. When in learning mode, this register could be read to see if an address has been learned.

|     |      |      |      |     |      |      |      |
|-----|------|------|------|-----|------|------|------|
| D7  | D6   | D5   | D4   | D3  | D2   | D1   | D0   |
| ADV | PTR2 | PTR1 | PTR0 | ADV | PTR2 | PTR1 | PTR0 |

| Bit | R/W | Symbol | Description                                                                                                             |
|-----|-----|--------|-------------------------------------------------------------------------------------------------------------------------|
| D0  | R   | PTR0   | D0 of the pointer for the port CAM location 1.                                                                          |
| D1  | R   | PTR1   | D1 of the pointer for the port CAM location 1.                                                                          |
| D2  | R   | PTR2   | D2 of the pointer for the port CAM location 1.                                                                          |
| D3  | R/W | ADV    | <b>Address Valid:</b><br>0: Address is not valid in port CAM location 1.<br>1: Address is valid in port CAM location 1. |
| D4  | R   | PTR0   | D0 of the pointer for the port CAM location 2.                                                                          |
| D5  | R   | PTR1   | D1 of the pointer for the port CAM location 2.                                                                          |
| D6  | R   | PTR2   | D2 of the pointer for the port CAM location 2.                                                                          |
| D7  | R/W | ADV    | <b>Address Valid:</b><br>0: Address is not valid in port CAM location 2.<br>1: Address is valid in port CAM location 2. |

### PORT CAM REGISTER (PAGES 5H, 6H, 8H, 9H)

This register accesses the 48 bits of the port CAM address. Six write/read cycles are required to load/read the entire 48-bit address.

|              |              |              |              |              |              |              |              |
|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| D7           | D6           | D5           | D4           | D3           | D2           | D1           | D0           |
| PCAMx<br>_D7 | PCAMx<br>_D6 | PCAMx<br>_D5 | PCAMx<br>_D4 | PCAMx<br>_D3 | PCAMx<br>_D2 | PCAMx<br>_D1 | PCAMx<br>_D0 |

| Bit    | R/W | Symbol           | Description                                                                                                                                                                                                                                                                                                                                                                               |
|--------|-----|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D(7:0) | R/W | PCAMx<br>_D(7:0) | This register access the <b>Port CAM</b> for the particular port. Note that <b>x</b> represents the port number.<br><br>1st access: bits [7:0] of the address,<br>2nd access: bits [15:8] of the address,<br>3rd access: bits [23:16] of the address,<br>4th access: bits [31:24] of the address,<br>5th access: bits [39:32] of the address,<br>6th access: bits [47:40] of the address. |

## 8.0 RIC II Registers (Continued)

### SHARED CAM VALIDATION REGISTER 1 (SCVR 1) (PAGE 9H ADDRESS 16H)

This register indicates the validity of an Ethernet address stored in any one of the shared CAMs. When a "1" is written in a specific SCVR, upon starting the network security, the CAM contents will be used for address comparison.

| D7   | D6   | D5   | D4   | D3   | D2   | D1   | D0   |
|------|------|------|------|------|------|------|------|
| ADV8 | ADV7 | ADV6 | ADV5 | ADV4 | ADV3 | ADV2 | ADV1 |

| Bit | R/W | Symbol | Description                                                                                   |
|-----|-----|--------|-----------------------------------------------------------------------------------------------|
| D0  | R/W | ADV1   | <b>Address Valid 1:</b><br>0: Address is not valid in CAM 1.<br>1: Address is valid in CAM 1. |
| D1  | R/W | ADV2   | <b>Address Valid 2:</b><br>0: Address is not valid in CAM 2.<br>1: Address is valid in CAM 2. |
| D2  | R/W | ADV3   | <b>Address Valid 3:</b><br>0: Address is not valid in CAM 3.<br>1: Address is valid in CAM 3. |
| D3  | R/W | ADV4   | <b>Address Valid 4:</b><br>0: Address is not valid in CAM 4.<br>1: Address is valid in CAM 4. |
| D4  | R/W | ADV5   | <b>Address Valid 5:</b><br>0: Address is not valid in CAM 5.<br>1: Address is valid in CAM 5. |
| D5  | R/W | ADV6   | <b>Address Valid 6:</b><br>0: Address is not valid in CAM 6.<br>1: Address is valid in CAM 6. |
| D6  | R/W | ADV7   | <b>Address Valid 7:</b><br>0: Address is not valid in CAM 7.<br>1: Address is valid in CAM 7. |
| D7  | R/W | ADV8   | <b>Address Valid 8:</b><br>0: Address is not valid in CAM 8.<br>1: Address is valid in CAM 8. |

**Note:** Before writing to and changing any bits in this register, read the register first and then only change the desired bits. By doing this, a previous entry will not be invalidated mistakenly.

## 8.0 RIC II Registers (Continued)

### SHARED CAM VALIDATION REGISTER 2 (SCVR 2) (PAGE 9H, ADDRESS 17H)

This register indicates the validity of an Ethernet address stored in any one of the shared CAMs. When a "1" is written in a specific SCVR, upon starting the network security, the CAM contents will be used for address comparison.

|       |       |       |       |       |       |       |      |
|-------|-------|-------|-------|-------|-------|-------|------|
| D7    | D6    | 5     | D4    | D3    | D2    | D1    | D0   |
| ADV16 | ADV15 | ADV14 | ADV13 | ADV12 | ADV11 | ADV10 | ADV9 |

| Bit | R/W | Symbol | Description                                                                                      |
|-----|-----|--------|--------------------------------------------------------------------------------------------------|
| D0  | R/W | ADV9   | <b>Address Valid 9:</b><br>0: Address is not valid in CAM 9.<br>1: Address is valid in CAM 9.    |
| D1  | R/W | ADV10  | <b>Address Valid 10:</b><br>0: Address is not valid in CAM 10.<br>1: Address is valid in CAM 10. |
| D2  | R/W | ADV11  | <b>Address Valid 11:</b><br>0: Address is not valid in CAM 11.<br>1: Address is valid in CAM 11. |
| D3  | R/W | ADV12  | <b>Address Valid 12:</b><br>0: Address is not valid in CAM 12.<br>1: Address is valid in CAM 12. |
| D4  | R/W | ADV13  | <b>Address Valid 13:</b><br>0: Address is not valid in CAM 13.<br>1: Address is valid in CAM 13. |
| D5  | R/W | ADV14  | <b>Address Valid 14:</b><br>0: Address is not valid in CAM 14.<br>1: Address is valid in CAM 14. |
| D6  | R/W | ADV15  | <b>Address Valid 15:</b><br>0: Address is not valid in CAM 15.<br>1: Address is valid in CAM 15. |
| D7  | R/W | ADV16  | <b>Address Valid 16:</b><br>0: Address is not valid in CAM 16.<br>1: Address is valid in CAM 16. |

**Note:** Before writing to and changing any bits in this register, read the register first and then only change the desired bits. By doing this, a previous entry will not be invalidated mistakenly.

## 8.0 RIC II Registers (Continued)

### SHARED CAM VALIDATION REGISTER 3 (SCVR 3) (PAGE 9H, ADDRESS 18H)

This register indicates the validity of an Ethernet address stored in any one of the shared CAMs. When a "1" is written in a specific SCVR, upon starting the network security, the CAM contents will be used for address comparison.

| D7    | D6    | 5     | D4    | D3    | D2    | D1    | D0    |
|-------|-------|-------|-------|-------|-------|-------|-------|
| ADV24 | ADV23 | ADV22 | ADV21 | ADV20 | ADV19 | ADV18 | ADV17 |

| Bit | R/W | Symbol | Description                                                                                      |
|-----|-----|--------|--------------------------------------------------------------------------------------------------|
| D0  | R/W | ADV17  | <b>Address Valid 17:</b><br>0: Address is not valid in CAM 17.<br>1: Address is valid in CAM 17. |
| D1  | R/W | ADV18  | <b>Address Valid 18:</b><br>0: Address is not valid in CAM 18.<br>1: Address is valid in CAM 18. |
| D2  | R/W | ADV19  | <b>Address Valid 19:</b><br>0: Address is not valid in CAM 19.<br>1: Address is valid in CAM 19. |
| D3  | R/W | ADV20  | <b>Address Valid 20:</b><br>0: Address is not valid in CAM 20.<br>1: Address is valid in CAM 20. |
| D4  | R/W | ADV21  | <b>Address Valid 21:</b><br>0: Address is not valid in CAM 21.<br>1: Address is valid in CAM 21. |
| D5  | R/W | ADV22  | <b>Address Valid 22:</b><br>0: Address is not valid in CAM 22.<br>1: Address is valid in CAM 22. |
| D6  | R/W | ADV23  | <b>Address Valid 23:</b><br>0: Address is not valid in CAM 23.<br>1: Address is valid in CAM 23. |
| D7  | R/W | ADV24  | <b>Address Valid 24:</b><br>0: Address is not valid in CAM 24.<br>1: Address is valid in CAM 24. |

**Note:** Before writing to and changing any bits in this register, read the register first and then only change the desired bits. By doing this, a previous entry will not be invalidated mistakenly.

## 8.0 RIC II Registers (Continued)

### SHARED CAM VALIDATION REGISTER 4 (SCVR 4) (PAGE 9H ADDRESS 19H)

This register indicates the validity of an Ethernet address stored in any one of the shared CAMs. When a "1" is written in a specific SCVR, upon starting the network security, the CAM contents will be used for address comparison.

| D7    | D6    | 5     | D4    | D3    | D2    | D1    | D0    |
|-------|-------|-------|-------|-------|-------|-------|-------|
| ADV32 | ADV31 | ADV30 | ADV29 | ADV28 | ADV27 | ADV26 | ADV25 |

| Bit | R/W | Symbol | Description                                                                                      |
|-----|-----|--------|--------------------------------------------------------------------------------------------------|
| D0  | R/W | ADV25  | <b>Address Valid 25:</b><br>0: Address is not valid in CAM 25.<br>1: Address is valid in CAM 25. |
| D1  | R/W | ADV26  | <b>Address Valid 26:</b><br>0: Address is not valid in CAM 26.<br>1: Address is valid in CAM 26. |
| D2  | R/W | ADV27  | <b>Address Valid 27:</b><br>0: Address is not valid in CAM 27.<br>1: Address is valid in CAM 27. |
| D3  | R/W | ADV28  | <b>Address Valid 28:</b><br>0: Address is not valid in CAM 28.<br>1: Address is valid in CAM 28. |
| D4  | R/W | ADV29  | <b>Address Valid 29:</b><br>0: Address is not valid in CAM 29.<br>1: Address is valid in CAM 29. |
| D5  | R/W | ADV30  | <b>Address Valid 30:</b><br>0: Address is not valid in CAM 30.<br>1: Address is valid in CAM 30. |
| D6  | R/W | ADV31  | <b>Address Valid 31:</b><br>0: Address is not valid in CAM 31.<br>1: Address is valid in CAM 31. |
| D7  | R/W | ADV32  | <b>Address Valid 32:</b><br>0: Address is not valid in CAM 32.<br>1: Address is valid in CAM 32. |

**Note:** Before writing to and changing any bits in this register, read the register first and then only change the desired bits. By doing this, a previous entry will not be invalidated mistakenly.

### 8.0 RIC II Registers (Continued)

#### SHARED CAM REGISTER (PAGES 9H, AH, BH, CH, DH, EH, FH)

This register accesses the 48 bits of the shared CAM address. Six write/read cycles are required to load/read the entire 48-bit address.

| D7            | D6            | D5            | D4            | D3            | D2            | D1            | D0            |
|---------------|---------------|---------------|---------------|---------------|---------------|---------------|---------------|
| SCAMx<br>__D7 | SCAMx<br>__D6 | SCAMx<br>__D5 | SCAMx<br>__D4 | SCAMx<br>__D3 | SCAMx<br>__D2 | SCAMx<br>__D1 | SCAMx<br>__D0 |

| Bit    | R/W | Symbol            | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|--------|-----|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D(7:0) | R/W | SCAMx<br>__D(7:0) | <p>This register access the <b>Shared CAM</b> for the particular port. Note that <b>x</b> represents the port number.</p> <p>1st access: bits [7:0] of the address,<br/>                     2nd access: bits [15:8] of the address,<br/>                     3rd access: bits [23:16] of the address,<br/>                     4th access: bits [31:24] of the address,<br/>                     5th access: bits [39:32] of the address,<br/>                     6th access: bits [47:40] of the address.</p> |

## 8.0 RIC II Registers (Continued)

### CAM LOCATION MASK REGISTER (CLMR) (PAGES 9H, AH, BH, CH, DH, EH, FH)

Each shared CAM has a CLMR, therefore there are 32 CLMRs. Any of the 32 CAMs can be shared among the ports. For example, multiple ports can share a single ethernet address, or multiple addresses can be associated with a single port. Assigning CAMs to ports, or vice-versa, is done through these registers.

#### CLMR Lo Byte Location

|    |    |    |    |    |    |    |    |
|----|----|----|----|----|----|----|----|
| D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 |
| P8 | P7 | P6 | P5 | P4 | P3 | P2 | P1 |

| Bit | R/W | Symbol | Description                                                                |
|-----|-----|--------|----------------------------------------------------------------------------|
| D0  | R/W | P1     | 0: CAM entry does not belong to port 1.<br>1: CAM entry belongs to port 1. |
| D1  | R/W | P2     | 0: CAM entry does not belong to port 2.<br>1: CAM entry belongs to port 2. |
| D2  | R/W | P3     | 0: CAM entry does not belong to port 3.<br>1: CAM entry belongs to port 3. |
| D3  | R/W | P4     | 0: CAM entry does not belong to port 4.<br>1: CAM entry belongs to port 4. |
| D4  | R/W | P5     | 0: CAM entry does not belong to port 5.<br>1: CAM entry belongs to port 5. |
| D5  | R/W | P6     | 0: CAM entry does not belong to port 6.<br>1: CAM entry belongs to port 6. |
| D6  | R/W | P7     | 0: CAM entry does not belong to port 7.<br>1: CAM entry belongs to port 7. |
| D7  | R/W | P8     | 0: CAM entry does not belong to port 8.<br>1: CAM entry belongs to port 8. |

#### CLMR Hi Byte Location

|      |      |      |     |     |     |     |    |
|------|------|------|-----|-----|-----|-----|----|
| D7   | D6   | D5   | D4  | D3  | D2  | D1  | D0 |
| PTR2 | PTR1 | PTR0 | P13 | P12 | P11 | P10 | P9 |

| Bit | R/W | Symbol | Description                                                                  |
|-----|-----|--------|------------------------------------------------------------------------------|
| D0  | R/W | P9     | 0: CAM entry does not belong to port 9.<br>1: CAM entry belongs to port 9.   |
| D1  | R/W | P10    | 0: CAM entry does not belong to port 10.<br>1: CAM entry belongs to port 10. |
| D2  | R/W | P11    | 0: CAM entry does not belong to port 11.<br>1: CAM entry belongs to port 11. |
| D3  | R/W | P12    | 0: CAM entry does not belong to port 12.<br>1: CAM entry belongs to port 12. |
| D4  | R/W | P13    | 0: CAM entry does not belong to port 13.<br>1: CAM entry belongs to port 13. |
| D5  | R   | PTR0   | D0 of the pointer into the CAM location.                                     |
| D6  | R   | PTR1   | D1 of the pointer into the CAM location.                                     |
| D7  | R   | PTR2   | D2 of the pointer into the CAM location.                                     |

## 9.0 AC and DC Specifications

### Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

|                                 |                          |
|---------------------------------|--------------------------|
| Supply Voltage ( $V_{CC}$ )     | 0.5V to 7.0V             |
| DC Input Voltage ( $V_{IN}$ )   | -0.5V to $V_{CC} + 0.5V$ |
| DC Output Voltage ( $V_{OUT}$ ) | -0.5V to $V_{CC} + 0.5V$ |

|                                                       |                 |
|-------------------------------------------------------|-----------------|
| Storage Temperature Range ( $T_{STG}$ )               | -65°C to +150°C |
| Power Dissipation ( $P_D$ )                           | 2W              |
| Lead Temperature ( $T_L$ )<br>(Soldering, 10 seconds) | 260°C           |
| ESD Rating<br>( $R_{zap} = 1.5k, C_{zap} = 120 pF$ )  | 1500V           |

### DC Specifications $T_A = 0^\circ C$ to $+70^\circ C$ , $V_{CC} = 5V \pm 5\%$ unless otherwise specified

| Symbol                                                                         | Description                                                          | Conditions                                          | Min                   | Max                    | Units   |
|--------------------------------------------------------------------------------|----------------------------------------------------------------------|-----------------------------------------------------|-----------------------|------------------------|---------|
| <b>PROCESSOR, LED, TWISTED PAIR PORTS, INTER-RIC AND MANAGEMENT INTERFACES</b> |                                                                      |                                                     |                       |                        |         |
| $V_{OH}$                                                                       | Minimum High Level Output Voltage                                    | $I_{OH} = -8 mA$                                    | 3.5                   |                        | V       |
| $V_{OL}$                                                                       | Minimum Low Level Output Voltage                                     | $I_{OL} = 8 mA$                                     |                       | 0.4                    | V       |
| $V_{IH}$                                                                       | Minimum High Level Input Voltage                                     |                                                     | 2.0                   |                        | V       |
| $V_{IL}$                                                                       | Maximum Low Level Input Voltage                                      |                                                     |                       | 0.8                    | V       |
| $I_{IN}$                                                                       | Input Current                                                        | $V_{IN} = V_{CC}$ or GND                            | -1.0                  | 1.0                    | $\mu A$ |
| $I_{OZ}$                                                                       | Maximum TRI-STATE Output Leakage Current                             | $V_{OUT} = V_{CC}$ or GND                           | -10                   | 10                     | $\mu A$ |
| $I_{CC}$                                                                       | Average Supply Current                                               | $V_{IN} = V_{CC}$ or GND, $V_{CC} = 5.25V$          |                       | 380                    | mA      |
| <b>AUI (PORT 1)</b>                                                            |                                                                      |                                                     |                       |                        |         |
| $V_{OD}$                                                                       | Differential Output Voltage ( $TX \pm$ )                             | 78 $\Omega$ Termination and 270 $\Omega$ Pulldowns  | $\pm 550$             | $\pm 1200$             | mV      |
| $V_{OB}$                                                                       | Differential Output Voltage Imbalance ( $TX \pm$ )                   | 78 $\Omega$ Termination and 270 $\Omega$ Pulldowns  | Typical: 40 mV        |                        |         |
| $V_U$                                                                          | Undershoot Voltage ( $TX \pm$ )                                      | 78 $\Omega$ Termination and 270 $\Omega$ Pulldowns  | Typical: 80 mV        |                        |         |
| $V_{DS}$                                                                       | Differential Squelch Threshold ( $RX \pm, CD \pm$ )                  |                                                     | -175                  | -300                   | mV      |
| $V_{CM}$                                                                       | Differential Input Common Mode Voltage ( $RX \pm, CD \pm$ ) (Note 1) |                                                     | 0                     | 5.5                    | V       |
| <b>PSEUDO AUI (PORTS 2-13)</b>                                                 |                                                                      |                                                     |                       |                        |         |
| $V_{POD}$                                                                      | Differential Output Voltage ( $TX \pm$ )                             | 270 $\Omega$ Termination and 1 k $\Omega$ Pulldowns | $\pm 450$             | $\pm 1200$             | mV      |
| $V_{POB}$                                                                      | Differential Output Voltage Imbalance ( $TX \pm$ )                   | 270 $\Omega$ Termination and 1 k $\Omega$ Pulldowns | Typical: 40 mV        |                        |         |
| $V_{PU}$                                                                       | Undershoot Voltage ( $TX \pm$ )                                      | 270 $\Omega$ Termination and 1 k $\Omega$ Pulldowns | Typical: 80 mV        |                        |         |
| $V_{PDS}$                                                                      | Differential Squelch Threshold ( $RX \pm, CD \pm$ )                  |                                                     | -175                  | -300                   | mV      |
| $V_{PCM}$                                                                      | Differential Input Common Mode Voltage ( $Rx \pm, CD \pm$ ) (Note 1) |                                                     | 0                     | 5.5                    | V       |
| <b>TWISTED PAIR (PORTS 2-13)</b>                                               |                                                                      |                                                     |                       |                        |         |
| $V_{RON}$                                                                      | Minimum Receive Squelch Threshold                                    | Normal Mode<br>Reduced Mode                         | $\pm 300$<br>(Note 2) | $\pm 585$<br>$\pm 340$ | mV      |

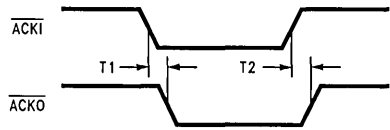
Note 1: This parameter is guaranteed by design and is not tested.

Note 2: The operation in Reduced Mode is not guaranteed below 300 mV.



# 10.0 Timing and Load Diagrams

## PORT ARBITRATION TIMING



TL/F/12499-24

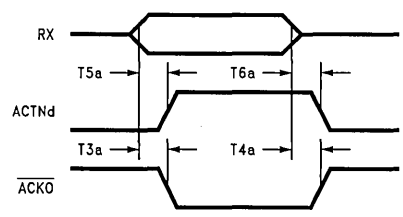
| Number | Symbol     | Parameter              | Min | Max | Units |
|--------|------------|------------------------|-----|-----|-------|
| T1     | ackilackol | ACKI Low to ACKO Low   |     | 24  | ns    |
| T2     | ackihackoh | ACKI High to ACKO High |     | 21  | ns    |

Note: Timing valid with no receive or collision activities.

Note: In these diagrams the Inter-RIC and Management Busses are shown using active high signals, active low signals may also be used. See Mode Load Operation description.

## RECEIVE TIMINGS—AUI PORTS

Receive activity propagation start up and end delays for ports in non 10BASE-T mode



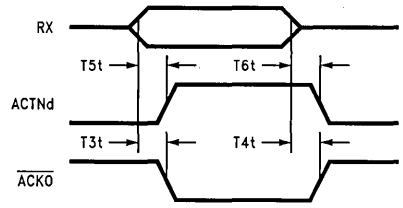
TL/F/12499-25

| Number | Symbol    | Parameter                     | Min | Max | Units |
|--------|-----------|-------------------------------|-----|-----|-------|
| T3a    | rxaaackol | RX Active to ACKO Low         |     | 66  | ns    |
| T4a    | rxiaackoh | RX Inactive to ACKO High      |     | 325 | ns    |
| T5a    | rxaaactna | RX Active to ACTNd Active     |     | 105 | ns    |
| T6a    | rxiaactni | RX Inactive to ACTNd Inactive |     | 325 | ns    |

Note: ACKI assumed high.

## RECEIVE TIMING—10BASE-T PORTS

Receive activity propagation start up and end delays for ports in 10BASE-T mode



TL/F/12499-26

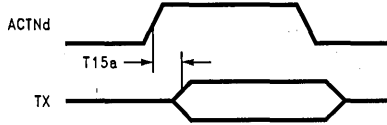
| Number | Symbol    | Parameter                     | Min | Max | Units |
|--------|-----------|-------------------------------|-----|-----|-------|
| T3t    | rxaaackol | RX Active to ACKO Low         |     | 240 | ns    |
| T4t    | rxiaackoh | RX Inactive to ACKO High      |     | 255 | ns    |
| T5t    | rxaaactna | RX Active to ACTNd Active     |     | 270 | ns    |
| T6t    | rxiaactni | RX Inactive to ACTNd Inactive |     | 265 | ns    |

Note: ACKI assumed high.

## 10.0 Timing and Load Diagrams (Continued)

### TRANSMIT TIMING—AUI PORTS

Transmit activity propagation start up and end delays for ports in non 10BASE-T mode



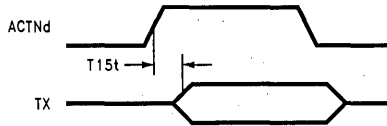
TL/F/12499-27

| Number | Symbol   | Parameter                 | Min | Max | Units |
|--------|----------|---------------------------|-----|-----|-------|
| T15a   | actnatxa | ACTNd Active to TX Active |     | 585 | ns    |

Note: ACKI assumed high.

### TRANSMIT TIMING—10BASE-T PORTS

Receive activity propagation start up and end delays for ports in 10BASE-T mode



TL/F/12499-28

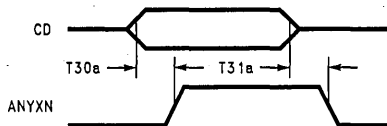
| Number | Symbol   | Parameter                 | Min | Max | Units |
|--------|----------|---------------------------|-----|-----|-------|
| T15t   | actnatxa | ACTNd Active to TX Active |     | 790 | ns    |

Note: ACKI assumed high.

### COLLISION TIMING—AUI PORTS

Collision activity propagation start up and end delays for ports in non 10BASE-T mode

#### TRANSMIT COLLISION TIMING



TL/F/12499-29

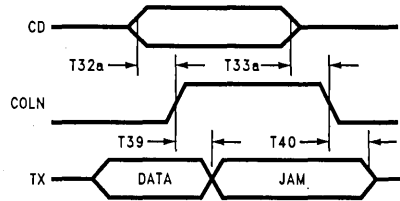
| Number | Symbol    | Parameter                                  | Min | Max | Units |
|--------|-----------|--------------------------------------------|-----|-----|-------|
| T30a   | cdaanyxna | CD Active to ANYXN Active                  |     | 65  | ns    |
| T31a   | cdianyxn  | CD Inactive to ANYXN Inactive (Notes 1, 2) |     | 400 | ns    |

Note 1: TX collision extension has already been performed and no other port is driving ANYXN.

Note 2: Includes TW2.

# 10.0 Timing and Load Diagrams (Continued)

## RECEIVE COLLISION TIMING



TL/F/12499-30

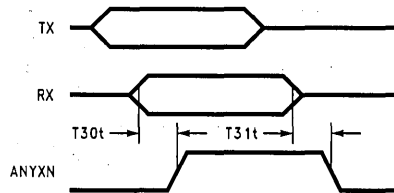
| Number | Symbol   | Parameter                            | Min | Max | Units |
|--------|----------|--------------------------------------|-----|-----|-------|
| T32a   | cdacolna | CD Active to COLN Active (Note 1)    |     | 55  | ns    |
| T33a   | cdicolni | CD Inactive to COLN Inactive         |     | 215 | ns    |
| T39    | colnajs  | COLN Active to Start of Jam          |     | 400 | ns    |
| T40    | colnije  | COLN Inactive to End of Jam (Note 2) |     | 800 | ns    |

Note 1: PKEN assumed high.

Note 2: Assuming reception ended before COLN goes inactive. TW2 is included in this parameter. Assuming ACTNd to ACTNs delay is 0.

## COLLISION TIMING—10BASE-T PORTS

Collision activity propagation start up and end delays for ports in 10BASE-T mode



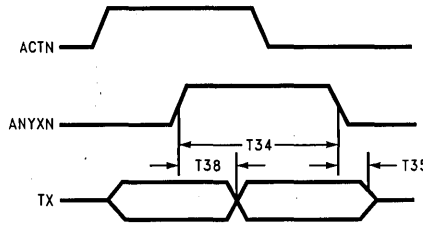
TL/F/12499-31

| Number | Symbol   | Parameter                                     | Min | Max | Units |
|--------|----------|-----------------------------------------------|-----|-----|-------|
| T30t   | colaanya | Collision Active to ANYXN Active              |     | 800 | ns    |
| T31t   | colianyi | Collision Inactive to ANYXN Inactive (Note 1) |     | 400 | ns    |

Note 1: TX collision extension has already been performed and no other port is asserting ANYXN.

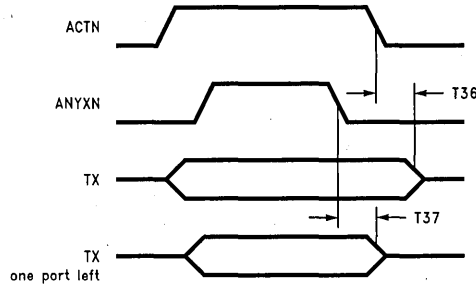
# 10.0 Timing and Load Diagrams (Continued)

## COLLISION TIMING—AUI PORTS



TL/F/12499-32

| Number | Symbol   | Parameter                            | Min | Max | Units |
|--------|----------|--------------------------------------|-----|-----|-------|
| T34    | anyamin  | ANYXN Active Time                    | 96  |     | Bits  |
| T35    | anyitxai | ANYXN Inactive to TX to all Inactive | 120 | 170 | ns    |
| T38    | anyasj   | ANYXN Active to Start of Jam         |     | 400 | ns    |

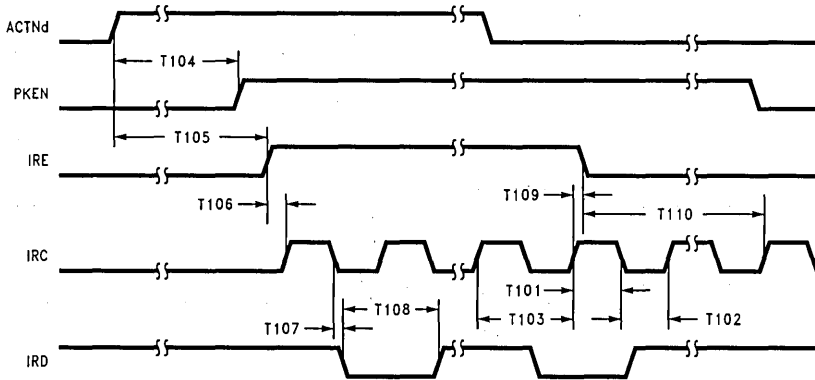


TL/F/12499-33

| Number | Symbol   | Parameter                                     | Min | Max | Units |
|--------|----------|-----------------------------------------------|-----|-----|-------|
| T36    | actnitxi | ACTN Inactive to TX Inactive                  |     | 405 | ns    |
| T37    | anyitxoi | ANYXN Inactive to TX "One Port Left" Inactive | 120 | 170 | ns    |

# 10.0 Timing and Load Diagrams (Continued)

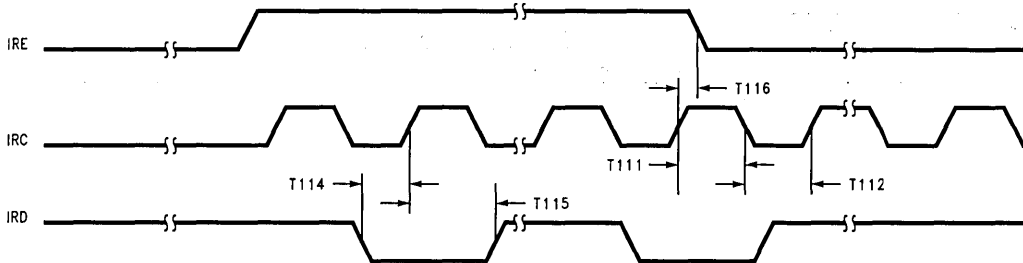
## INTER RIC BUS OUTPUT TIMING



TL/F/12499-34

| Number | Symbol      | Parameter                         | Min | Max | Units |
|--------|-------------|-----------------------------------|-----|-----|-------|
| T101   | ircoh       | IRC Output High Time              | 45  | 55  | ns    |
| T102   | ircol       | IRC Output Low Time               | 45  | 55  | ns    |
| T103   | ircoc       | IRC Output Cycle Time             | 90  | 110 | ns    |
| T104   | actndapkena | ACTNd Active to PKEN Active       | 555 |     | ns    |
| T105   | actndairea  | ACTNd Active to IRE Active        | 560 |     | ns    |
| T106   | ireairca    | IRE Output Active to IRC Active   |     | 1.8 | μs    |
| T107   | irdov       | IRD Output Valid from IRC         |     | 10  | ns    |
| T108   | irdos       | IRD Output Stable Valid Time      | 90  |     | ns    |
| T109   | ircohirei   | IRC Output High to IRE Inactive   | 30  | 70  | ns    |
| T110   | ircclks     | Number of IRCs after IRE Inactive | 5   |     | clks  |

## INTER RIC BUS INPUT TIMING

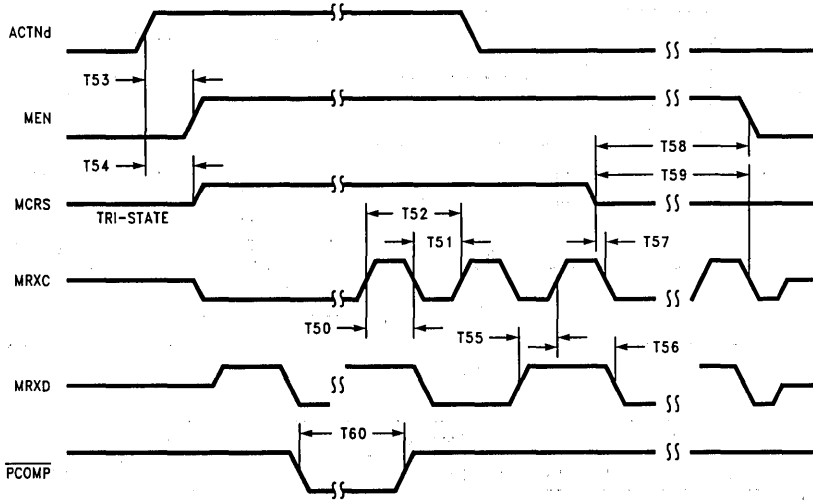


TL/F/12499-35

| Number | Symbol    | Parameter                      | Min | Max | Units |
|--------|-----------|--------------------------------|-----|-----|-------|
| T111   | ircih     | IRC Input High Time            | 20  |     | ns    |
| T112   | ircil     | IRC Input Low Time             | 20  |     | ns    |
| T114   | irdisirc  | IRD Input Setup to IRC         | 5   |     | ns    |
| T115   | irdihirc  | IRD Input Hold from IRC        | 10  |     | ns    |
| T116   | ircihirei | IRC Input High to IRE Inactive | 10  | 90  | ns    |

# 10.0 Timing and Load Diagrams (Continued)

## MANAGEMENT BUS TIMING



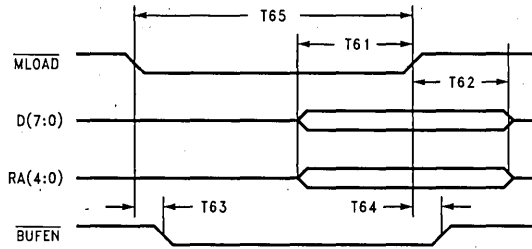
TL/F/12499-36

| Number | Symbol      | Parameter                               | Min | Max | Units |
|--------|-------------|-----------------------------------------|-----|-----|-------|
| T50    | mrch        | MRXC High Time                          | 45  | 55  | ns    |
| T51    | mrcl        | MRXC Low Time                           | 45  | 55  | ns    |
| T52    | mrxcd       | MRXC Cycle Time                         | 90  | 110 | ns    |
| T53    | actndamena  | ACTNd Active to MEN Active              |     | 715 | ns    |
| T54    | actndamcrsa | ACTNd Active to MCRS Active             |     | 720 | ns    |
| T55    | mrlds       | MRXD Setup                              | 40  |     | ns    |
| T56    | mrldh       | MRXD Hold                               | 45  |     | ns    |
| T57    | mrclmcrsi   | MRXC Low to MCRS Inactive               | -5  | 6   | ns    |
| T58    | mcrsimenl   | MCRS Inactive to MEN Low                |     | 510 | ns    |
| T59    | mrxcclks    | Min Number of MRXCs after MCRS Inactive | 5   | 5   | clks  |
| T60    | pcompw      | PCOMP Pulse Width                       | 20  |     | ns    |

Note: The preamble on this bus consists of the following string; 01011.

# 10.0 Timing and Load Diagrams (Continued)

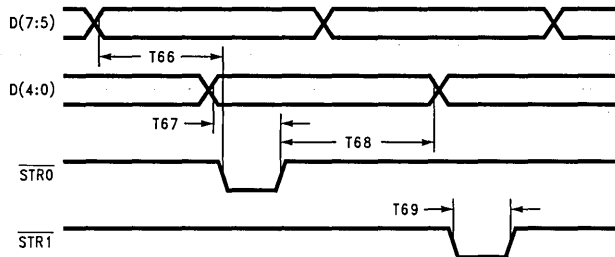
## MLOAD TIMING



TL/F/12499-37

| Number | Symbol  | Parameter                        | Min | Max | Units |
|--------|---------|----------------------------------|-----|-----|-------|
| T61    | mldats  | Data Setup                       | 10  |     | ns    |
| T62    | mldath  | Data Hold                        | 10  |     | ns    |
| T63    | mlabufa | MLOAD Active to BUFEN Active     |     | 35  | ns    |
| T64    | mlibufi | MLOAD Inactive to BUFEN Inactive |     | 35  | ns    |
| T65    | mlw     | MLOAD Width                      | 800 |     | ns    |

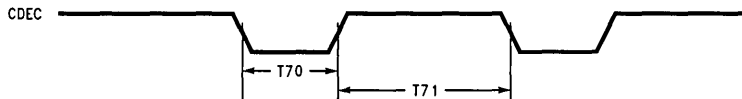
## STROBE TIMING



TL/F/12499-38

| Number | Symbol  | Parameter            | Min | Max | Units |
|--------|---------|----------------------|-----|-----|-------|
| T66    | strads  | Strobe Address Setup | 80  | 115 | ns    |
| T67    | strdats | Strobe Data Setup    | 40  | 65  | ns    |
| T68    | strdath | Strobe Data Hold     | 135 | 160 | ns    |
| T69    | strw    | Strobe Width         | 30  | 65  | ns    |

## CDEC TIMING

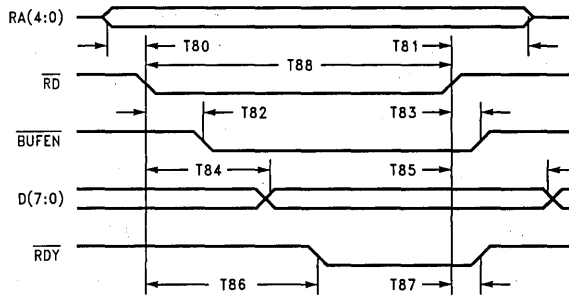


TL/F/12499-39

| Number | Symbol  | Parameter          | Min | Max | Units |
|--------|---------|--------------------|-----|-----|-------|
| T70    | cdecpw  | CDEC Pulse Width   | 20  | 100 | ns    |
| T71    | cdecdec | CDEC to CDEC Width | 200 |     | ns    |

# 10.0 Timing and Load Diagrams (Continued)

## REGISTER READ TIMING



TL/F/12499-40

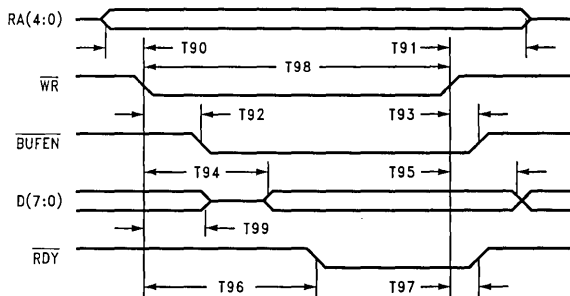
| Number | Symbol  | Parameter                       | Min | Max | Units |
|--------|---------|---------------------------------|-----|-----|-------|
| T80    | rdadrs  | Read Address Setup              | 0   |     | ns    |
| T81    | rdadrh  | Read Address Hold               | 0   |     | ns    |
| T82    | rdabufa | Read Active to BUFEN Active     | 95  | 345 | ns    |
| T83    | rdibufi | Read Inactive to BUFEN Inactive |     | 35  | ns    |
| T84    | rdadatv | Read Active to Data Valid       | 245 |     | ns    |
| T85    | rddath  | Read Data Hold                  | 75  |     | ns    |
| T86    | rdarda  | Read Active to RDY Active       | 340 | 585 | ns    |
| T87    | rdirdyi | Read Inactive to RDY Inactive   |     | 30  | ns    |
| T88    | rdw     | Read Width                      | 600 |     | ns    |

Note: Minimum high time between read/write cycles is 100 ns.



# 10.0 Timing and Load Diagrams (Continued)

## REGISTER WRITE TIMING



TL/F/12499-41

| Number | Symbol  | Parameter                                            | Min | Max | Units |
|--------|---------|------------------------------------------------------|-----|-----|-------|
| T90    | wradrs  | Write Address Setup                                  | 0   |     | ns    |
| T91    | wradrh  | Write Address Hold                                   | 0   |     | ns    |
| T92    | wrabufa | Write Active to $\overline{\text{BUFEN}}$ Active     | 95  | 355 | ns    |
| T93    | wribufi | Write Inactive to $\overline{\text{BUFEN}}$ Inactive |     | 35  | ns    |
| T94    | wradatv | Write Active to Data Valid                           |     | 275 | ns    |
| T95    | wrdath  | Write Data Hold                                      | 0   |     | ns    |
| T96    | wrardya | Write Active to $\overline{\text{RDY}}$ Active       | 340 | 585 | ns    |
| T97    | wrirdyi | Write Inactive to $\overline{\text{RDY}}$ Inactive   |     | 30  | ns    |
| T98    | wrw     | Write Width                                          | 600 |     | ns    |
| T99    | wradt   | Write Active to Data TRI-STATE                       |     | 350 | ns    |

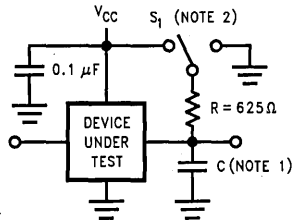
Note: Assuming zero propagation delay on external buffer.

Note: Minimum high time between read/write cycles is 100 ns.

## 11.0 AC Timing Test Conditions

All specifications are valid only if the mandatory isolation is employed and all differential signals are taken to be at the AUI side of the pulse transformer.

|                                              |                                 |
|----------------------------------------------|---------------------------------|
| Input Pulse Levels (TTL/CMOS)                | GND to 3.0V                     |
| Input Rise and Fall Times (TTL/CMOS)         | 5 ns                            |
| Input and Output Reference Levels (TTL/CMOS) | 1.5V                            |
| Input Pulse Levels (Diff.)                   | 2.0 V <sub>p-p</sub>            |
| Input and Output Reference Levels (Diff.)    | 50% Point of the Differential   |
| TRI-STATE Reference Levels                   | Float ( $\Delta V$ ) $\pm 0.5V$ |
| Output Load (See <i>Figure Below</i> )       |                                 |



TL/F/12499-42

**Note 1:** 100 pF, includes scope and jig capacitance.

**Note 2:** S1 = Open for timing tests for push pull outputs.

S1 = V<sub>CC</sub> for V<sub>OL</sub> test.

S1 = GND for V<sub>OH</sub> test.

S1 = V<sub>CC</sub> for High Impedance to active low and active low to High Impedance measurements.

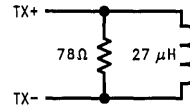
S1 = GND for High Impedance to active high and active high to High Impedance measurements.

### Capacitance $T_A = 25^\circ C, f = 1 \text{ MHz}$

| Symbol           | Parameter          | Typ | Units |
|------------------|--------------------|-----|-------|
| C <sub>IN</sub>  | Input Capacitance  | 7   | pF    |
| C <sub>OUT</sub> | Output Capacitance | 7   | pF    |

### DERATING FACTOR

Output timings are measured with a purely capacitive load for 50 pF. The following correction factor can be used for other loads:  $C_L \geq 50 \text{ pF} + 0.3 \text{ ns/pF}$ .



TL/F/12499-43

**Note:** In the above diagram, the TX+ and TX- signals are taken from the AUI side of the isolation (pulse transformer). The pulse transformer used for all testing is the Pulse Engineering PE64103.

# DP83955A/DP83956A LERIC™ LitE Repeater Interface Controller

## General Description

The DP83955/56 LitE Repeater Interface Controller (LERIC) may be used to implement an IEEE 802.3 multiport repeater unit. It fully satisfies the IEEE 802.3 repeater specification including the functions defined by the repeater, segment partition and jabber lockup protection state machines.

The LERIC has an on-chip phase-locked-loop (PLL) for Manchester data decoding, a Manchester encoder, and an Elasticity Buffer for preamble regeneration.

Each LERIC can connect up to 7 cable segments via its network interface ports. One port is fully Attachment Unit Interface (AUI) compatible and is able to connect to an external Medium Attachment Unit (MAU) using the maximum length of AUI cable. The other 6 ports have integrated 10BASE-T transceivers. These transceiver functions may be bypassed so that the LERIC may be used with external transceivers, such as National's DP8392 coaxial transceiver. In addition, large repeater units may be constructed by cascading LERICs together over the Inter-LERIC™ or Inter-RIC™ bus.

The LERIC is configurable for specific applications. It provides port status information for LED array displays. Additionally, the LERIC has a  $\mu$ P interface to provide individual port status, configuration, and port enable/disable functions.

The DP83956 has all the features of the DP83955, except that two of the bidirectional signals on DP83955 are changed to unidirectional signals on DP83956, and one more signal is added to DP83956 to accommodate the addition of bus transceivers for cascading a greater number of LERICs in large repeater applications.

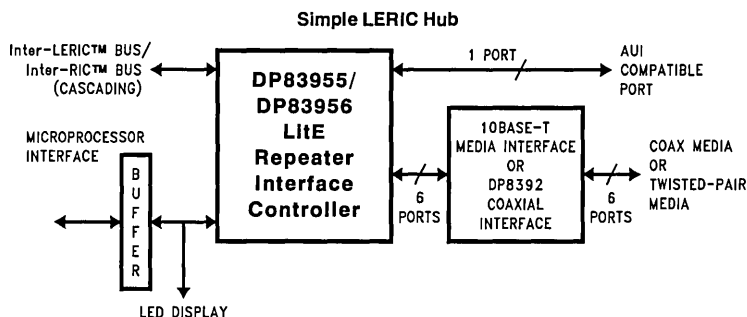
Specifications enclosed describe both the DP83955 and the DP83956 unless otherwise noted.

For IEEE 802.3 multiport repeater applications which require conformance to the IEEE 802.3 Draft Repeater Management options, the DP83950 Repeater Interface Controller (RIC™) is recommended especially for highly-managed hub requirements.

## Features

- Compliant with the IEEE 802.3 Repeater Specification
- 7 network connections (ports) per chip
- Selectable on-chip twisted-pair transceivers
- Cascadable for large multiple RIC/LERIC hub applications
- Compatible with AUI compliant transceivers
- On-chip Elasticity Buffer, Manchester encoder and decoder
- Separation Partition state machines for each port
- Provides port status information for LED displays including: receive, collision, partition, polarity, and link status
- Power-up configuration options—Repeater and Partition Specifications, Transceiver Interface, Status Display, Processor Operations
- Simple processor interface for repeater management and port disable
- Per port receive squelch level selection
- CMOS process for low power dissipation
- Single 5V supply

## 1.0 System Diagram



TL/F/11240-1

## Table of Contents

### 1.0 SYSTEM DIAGRAM

### 2.0 CONNECTION DIAGRAMS

### 3.0 PIN DESCRIPTION

### 4.0 BLOCK DIAGRAM

### 5.0 FUNCTIONAL DESCRIPTION

- 5.1 Overview of LERIC Functions
- 5.2 Description of Repeater Operations
- 5.3 Examples of Packet Repetition Scenarios
- 5.4 Description of Hardware Connection for Cascading
- 5.5 Processor and Display Interface
- 5.6 Processor and Display Interface Hardware Connection

### 6.0 PORT BLOCK FUNCTIONS

- 6.1 Transceiver Functions
- 6.2 Segment Partition
- 6.3 Port Status Register Functions

### 7.0 RIC REGISTER DESCRIPTIONS

- 7.1 LERIC Register Address Map
- 7.2 LERIC Status Register
- 7.3 Port Status and Configuration Registers

### 8.0 ABSOLUTE MAXIMUM RATINGS

### 9.0 DC ELECTRICAL CHARACTERISTICS

### 10.0 SWITCHING CHARACTERISTICS

### 11.0 AC TIMING TEST CONDITIONS

### 12.0 PHYSICAL DIMENSIONS

## 2.0 Connection Diagrams

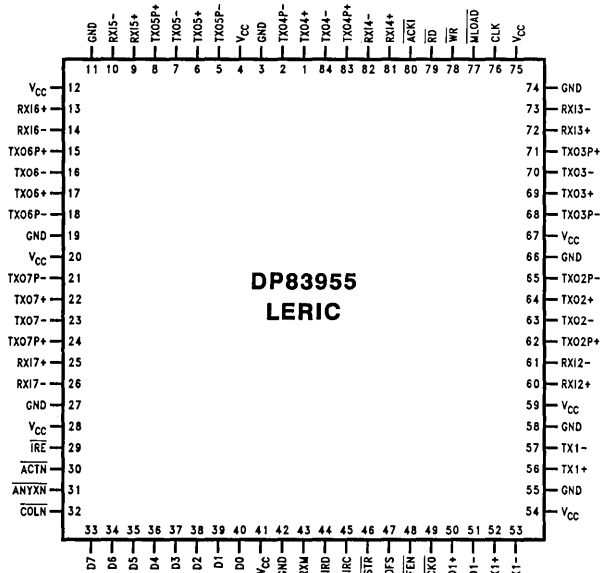
Pin Table for DP83955  
(Configured as Port 1 Full AUI, and Ports 2-7 Twisted-Pair)

| Pin Name        | Pin No. |
|-----------------|---------|
| TXO4+           | 1       |
| TXO4P-          | 2       |
| GND             | 3       |
| V <sub>CC</sub> | 4       |
| TXO5P-          | 5       |
| TXO5+           | 6       |
| TXO5-           | 7       |
| TXO5P+          | 8       |
| RXI5+           | 9       |
| RXI5-           | 10      |
| GND             | 11      |
| V <sub>CC</sub> | 12      |
| RXI6+           | 13      |
| RXI6-           | 14      |
| TXO6P+          | 15      |
| TXO6-           | 16      |
| TXO6+           | 17      |
| TXO6P-          | 18      |
| GND             | 19      |
| V <sub>CC</sub> | 20      |
| TXO7P-          | 21      |

| Pin Name        | Pin No. |
|-----------------|---------|
| TXO7+           | 22      |
| TXO7-           | 23      |
| TXO7P+          | 24      |
| RXI7+           | 25      |
| RXI7-           | 26      |
| GND             | 27      |
| V <sub>CC</sub> | 28      |
| IRE             | 29      |
| ACTN            | 30      |
| ANYXN           | 31      |
| COLN            | 32      |
| D7              | 33      |
| D6              | 34      |
| D5              | 35      |
| D4              | 36      |
| D3              | 37      |
| D2              | 38      |
| D1              | 39      |
| D0              | 40      |
| V <sub>CC</sub> | 41      |
| GND             | 42      |

| Pin Name        | Pin No. |
|-----------------|---------|
| RXM             | 43      |
| IRD             | 44      |
| IRC             | 45      |
| STR             | 46      |
| DFS             | 47      |
| BUFEN           | 48      |
| ACKO            | 49      |
| CD1+            | 50      |
| CD1-            | 51      |
| RX1+            | 52      |
| RX1-            | 53      |
| V <sub>CC</sub> | 54      |
| GND             | 55      |
| TX1+            | 56      |
| TX1-            | 57      |
| GND             | 58      |
| V <sub>CC</sub> | 59      |
| RXI2+           | 60      |
| RXI2-           | 61      |
| TXO2P+          | 62      |
| TXO2-           | 63      |

| Pin Name        | Pin No. |
|-----------------|---------|
| TXO2+           | 64      |
| TXO2P-          | 65      |
| GND             | 66      |
| V <sub>CC</sub> | 67      |
| TXO3P-          | 68      |
| TXO3+           | 69      |
| TXO3-           | 70      |
| TXO3P+          | 71      |
| RXI3+           | 72      |
| RXI3-           | 73      |
| GND             | 74      |
| V <sub>CC</sub> | 75      |
| CLK             | 76      |
| MLOAD           | 77      |
| WR              | 78      |
| RD              | 79      |
| ACKI            | 80      |
| RXI4+           | 81      |
| RXI4-           | 82      |
| TXO4P+          | 83      |
| TXO4-           | 84      |



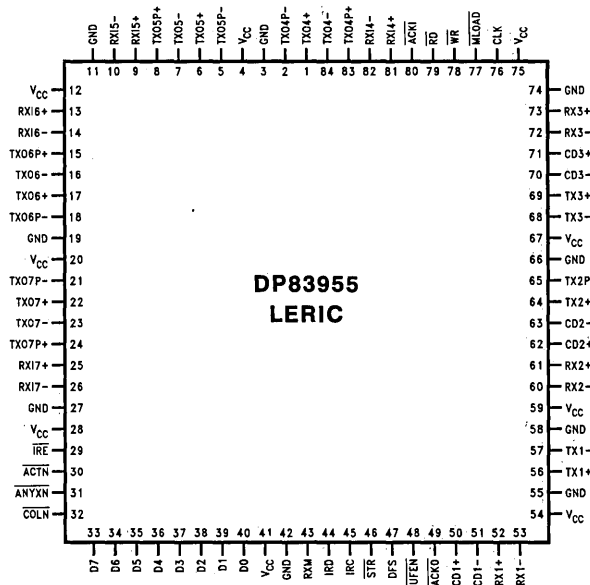
Top View

TL/F/11240-2

## 2.0 Connection Diagrams (Continued)

**Pin Table for DP83955**  
(Configured as Port 1 Full AUI, Ports 2-3 AUI, and Ports 4-7 Twisted-Pair)

| Pin Name        | Pin No. | Pin Name        | Pin No. | Pin Name        | Pin No. | Pin Name        | Pin No. |
|-----------------|---------|-----------------|---------|-----------------|---------|-----------------|---------|
| TXO4+           | 1       | TXO7+           | 22      | RXM             | 43      | TX2+            | 64      |
| TXO4P-          | 2       | TXO7-           | 23      | IRD             | 44      | TX2-            | 65      |
| GND             | 3       | TXO7P+          | 24      | IRC             | 45      | GND             | 66      |
| V <sub>CC</sub> | 4       | RXI7+           | 25      | STR             | 46      | V <sub>CC</sub> | 67      |
| TXO5P-          | 5       | RXI7-           | 26      | DFS             | 47      | TX3-            | 68      |
| TXO5+           | 6       | GND             | 27      | BUFEN           | 48      | TX3+            | 69      |
| TXO5-           | 7       | V <sub>CC</sub> | 28      | ACKO            | 49      | CD3-            | 70      |
| TXO5P+          | 8       | IRE             | 29      | CD1+            | 50      | CD3+            | 71      |
| RXI5+           | 9       | ACTN            | 30      | CD1-            | 51      | RX3-            | 72      |
| RXI5-           | 10      | ANYXN           | 31      | RX1+            | 52      | RX3+            | 73      |
| GND             | 11      | COLN            | 32      | RX1-            | 53      | GND             | 74      |
| V <sub>CC</sub> | 12      | D7              | 33      | V <sub>CC</sub> | 54      | V <sub>CC</sub> | 75      |
| RXI6+           | 13      | D6              | 34      | GND             | 55      | CLK             | 76      |
| RXI6-           | 14      | D5              | 35      | TX1+            | 56      | MLOAD           | 77      |
| TXO6P+          | 15      | D4              | 36      | TX1-            | 57      | WR              | 78      |
| TXO6-           | 16      | D3              | 37      | GND             | 58      | RD              | 79      |
| TXO6+           | 17      | D2              | 38      | V <sub>CC</sub> | 59      | ACKI            | 80      |
| TXO6P-          | 18      | D1              | 39      | RX2-            | 60      | RXI4+           | 81      |
| GND             | 19      | D0              | 40      | RX2+            | 61      | RXI4-           | 82      |
| V <sub>CC</sub> | 20      | V <sub>CC</sub> | 41      | CD2+            | 62      | TXO4P+          | 83      |
| TXO7P-          | 21      | GND             | 42      | CD2-            | 63      | TXO4-           | 84      |



**DP83955  
LERIC**

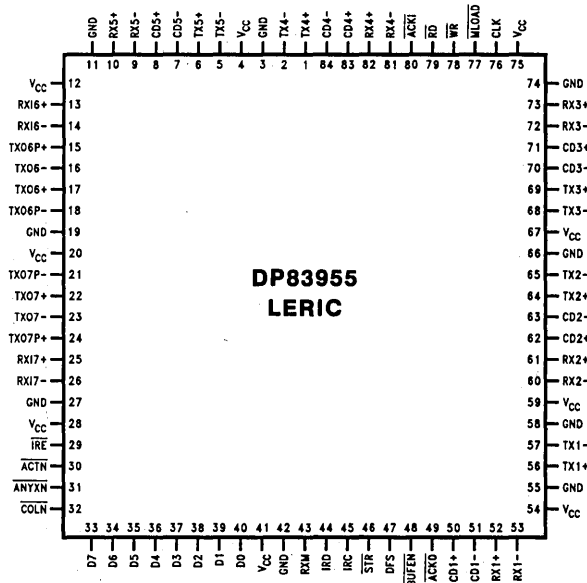
Top View

TL/F/11240-3

## 2.0 Connection Diagrams (Continued)

**Pin Table for DP83955**  
(Configured as Port 1 Full AUI, Ports 2-5 AUI, and Ports 6-7 Twisted-Pair)

| Pin Name        | Pin No. | Pin Name        | Pin No. | Pin Name        | Pin No. | Pin Name        | Pin No. |
|-----------------|---------|-----------------|---------|-----------------|---------|-----------------|---------|
| TX4+            | 1       | TXO7+           | 22      | RXM             | 43      | TX2+            | 64      |
| TX4-            | 2       | TXO7-           | 23      | IRD             | 44      | TX2-            | 65      |
| GND             | 3       | TXO7P+          | 24      | IRC             | 45      | GND             | 66      |
| V <sub>CC</sub> | 4       | RXI7+           | 25      | STR             | 46      | V <sub>CC</sub> | 67      |
| TX5-            | 5       | RXI7-           | 26      | DFS             | 47      | TX3-            | 68      |
| TX5+            | 6       | GND             | 27      | BUFEN           | 48      | TX3+            | 69      |
| CD5-            | 7       | V <sub>CC</sub> | 28      | ACKO            | 49      | CD3-            | 70      |
| CD5+            | 8       | IRE             | 29      | CD1+            | 50      | CD3+            | 71      |
| RX5-            | 9       | ACTN            | 30      | CD1-            | 51      | RX3-            | 72      |
| RX5+            | 10      | ANYXN           | 31      | RX1+            | 52      | RX3+            | 73      |
| GND             | 11      | COLN            | 32      | RX1-            | 53      | GND             | 74      |
| V <sub>CC</sub> | 12      | D7              | 33      | V <sub>CC</sub> | 54      | V <sub>CC</sub> | 75      |
| RXI6+           | 13      | D6              | 34      | GND             | 55      | CLK             | 76      |
| RXI6-           | 14      | D5              | 35      | TX1+            | 56      | MLOAD           | 77      |
| TXO6P+          | 15      | D4              | 36      | TX1-            | 57      | WR              | 78      |
| TXO6-           | 16      | D3              | 37      | GND             | 58      | RD              | 79      |
| TXO6+           | 17      | D2              | 38      | V <sub>CC</sub> | 59      | ACKI            | 80      |
| TXO6P-          | 18      | D1              | 39      | RX2-            | 60      | RX4-            | 81      |
| GND             | 19      | D0              | 40      | RX2+            | 61      | RX4+            | 82      |
| V <sub>CC</sub> | 20      | V <sub>CC</sub> | 41      | CD2+            | 62      | CD4+            | 83      |
| TXO7P-          | 21      | GND             | 42      | CD2-            | 63      | CD4-            | 84      |



**DP83955  
LERIC**

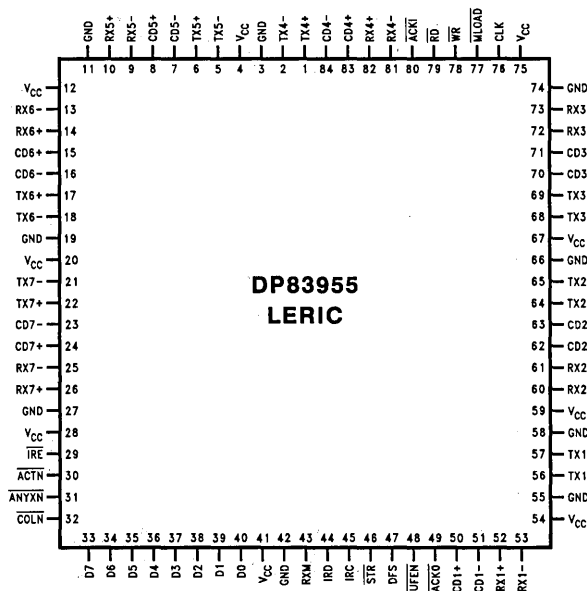
**Top View**

TL/F/11240-4

## 2.0 Connection Diagrams (Continued)

**Pin Table for DP83955**  
(Configured as Port 1 Full AUI, Ports 2-7 AUI)

| Pin Name        | Pin No. | Pin Name        | Pin No. | Pin Name        | Pin No. | Pin Name        | Pin No. |
|-----------------|---------|-----------------|---------|-----------------|---------|-----------------|---------|
| TX4+            | 1       | TX7+            | 22      | RXM             | 43      | TX2+            | 64      |
| TX4-            | 2       | CD7-            | 23      | IRD             | 44      | TX2-            | 65      |
| GND             | 3       | CD7+            | 24      | IRC             | 45      | GND             | 66      |
| V <sub>CC</sub> | 4       | RX7-            | 25      | STR             | 46      | V <sub>CC</sub> | 67      |
| TX5-            | 5       | RX7+            | 26      | DFS             | 47      | TX3-            | 68      |
| TX5+            | 6       | GND             | 27      | BUFEN           | 48      | TX3+            | 69      |
| CD5-            | 7       | V <sub>CC</sub> | 28      | ACKO            | 49      | CD3-            | 70      |
| CD5+            | 8       | IRE             | 29      | CD1+            | 50      | CD3+            | 71      |
| RX5-            | 9       | ACTN            | 30      | CD1-            | 51      | RX3-            | 72      |
| RX5+            | 10      | ANYXN           | 31      | RX1+            | 52      | RX3+            | 73      |
| GND             | 11      | COLN            | 32      | RX1-            | 53      | GND             | 74      |
| V <sub>CC</sub> | 12      | D7              | 33      | V <sub>CC</sub> | 54      | V <sub>CC</sub> | 75      |
| RX6-            | 13      | D6              | 34      | GND             | 55      | CLK             | 76      |
| RX6+            | 14      | D5              | 35      | TX1+            | 56      | MLOAD           | 77      |
| CD6+            | 15      | D4              | 36      | TX1-            | 57      | WR              | 78      |
| CD6-            | 16      | D3              | 37      | GND             | 58      | RD              | 79      |
| TX6+            | 17      | D2              | 38      | V <sub>CC</sub> | 59      | ACKI            | 80      |
| TX6-            | 18      | D1              | 39      | RX2-            | 60      | RX4-            | 81      |
| GND             | 19      | D0              | 40      | RX2+            | 61      | RX4+            | 82      |
| V <sub>CC</sub> | 20      | V <sub>CC</sub> | 41      | CD2+            | 62      | CD4+            | 83      |
| TX7-            | 21      | GND             | 42      | CD2-            | 63      | CD4-            | 84      |



Top View

TL/F/11240-5

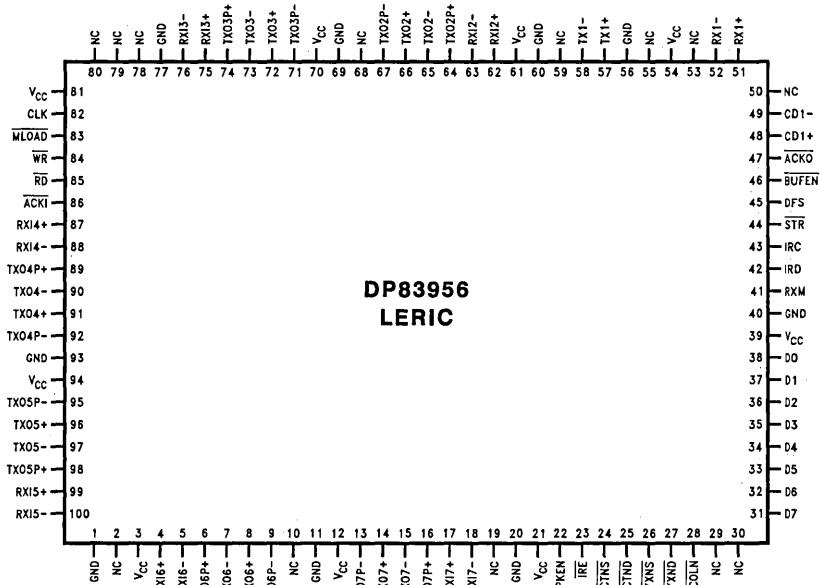


## 2.0 Connection Diagrams (Continued)

**Pin Table for DP83956**  
(Configured as Port 1 Full AUI, Ports 2-7 Twisted-Pair)

| Pin Name        | Pin No. | Pin Name            | Pin No. | Pin Name           | Pin No. | Pin Name        | Pin No. | Pin Name           | Pin No. |
|-----------------|---------|---------------------|---------|--------------------|---------|-----------------|---------|--------------------|---------|
| GND             | 1       | V <sub>CC</sub>     | 21      | RXM                | 41      | V <sub>CC</sub> | 61      | V <sub>CC</sub>    | 81      |
| NC              | 2       | PKEN                | 22      | IRD                | 42      | RXI2+           | 62      | CLK                | 82      |
| V <sub>CC</sub> | 3       | $\overline{IRE}$    | 23      | IRC                | 43      | RXI2-           | 63      | $\overline{MLOAD}$ | 83      |
| RXI6+           | 4       | $\overline{ACTNS}$  | 24      | $\overline{STR}$   | 44      | TXO2P+          | 64      | $\overline{WR}$    | 84      |
| RXI6-           | 5       | $\overline{ACTND}$  | 25      | DFS                | 45      | TXO2-           | 65      | $\overline{RD}$    | 85      |
| TXO6P+          | 6       | $\overline{ANYXNS}$ | 26      | $\overline{BUFEN}$ | 46      | TXO2+           | 66      | $\overline{ACKI}$  | 86      |
| TXO6-           | 7       | $\overline{ANYXND}$ | 27      | $\overline{ACKO}$  | 47      | TXO2P-          | 67      | RXI4+              | 87      |
| TXO6+           | 8       | $\overline{COLN}$   | 28      | CD1+               | 48      | NC              | 68      | RXI4-              | 88      |
| TXO6P-          | 9       | NC                  | 29      | CD1-               | 49      | GND             | 69      | TXO4P+             | 89      |
| NC              | 10      | NC                  | 30      | NC                 | 50      | V <sub>CC</sub> | 70      | TXO4-              | 90      |
| GND             | 11      | D7                  | 31      | RX1+               | 51      | TXO3P-          | 71      | TXO4+              | 91      |
| V <sub>CC</sub> | 12      | D6                  | 32      | RX1-               | 52      | TXO3+           | 72      | TXO4P-             | 92      |
| TXO7P-          | 13      | D5                  | 33      | NC                 | 53      | TXO3-           | 73      | GND                | 93      |
| TXO7+           | 14      | D4                  | 34      | V <sub>CC</sub>    | 54      | TXO3P+          | 74      | V <sub>CC</sub>    | 94      |
| TXO7-           | 15      | D3                  | 35      | NC                 | 55      | RXI3+           | 75      | TXO5P-             | 95      |
| TXO7P+          | 16      | D2                  | 36      | GND                | 56      | RXI3-           | 76      | TXO5+              | 96      |
| RXI7+           | 17      | D1                  | 37      | TX1+               | 57      | GND             | 77      | TXO5-              | 97      |
| RXI7-           | 18      | D0                  | 38      | TX1-               | 58      | NC              | 78      | TXO5P+             | 98      |
| NC              | 19      | V <sub>CC</sub>     | 39      | NC                 | 59      | NC              | 79      | RXI5+              | 99      |
| GND             | 20      | GND                 | 40      | GND                | 60      | NC              | 80      | RXI5-              | 100     |

Note: DP83956 will change from VLY package to VLJ package approximately Q3, 1993.



**DP83956**  
**LERIC**

Top View

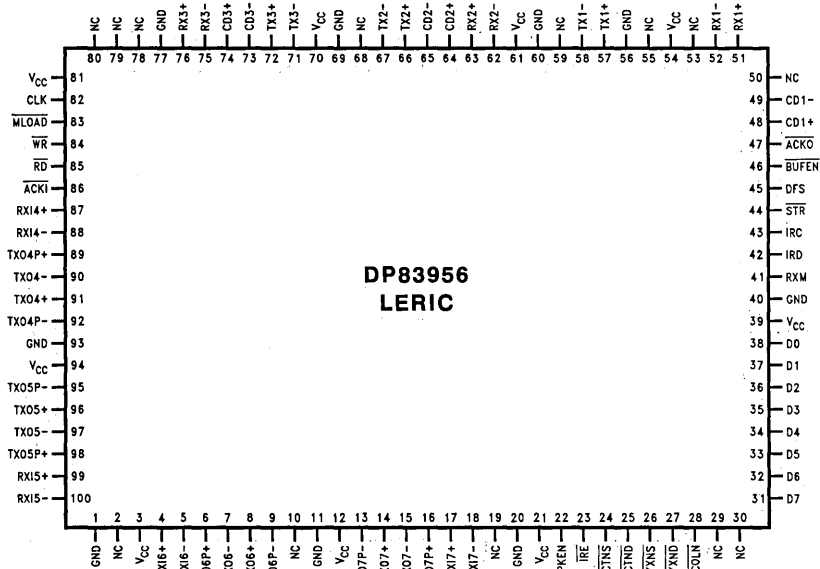
TL/F/11240-36

## 2.0 Connection Diagrams (Continued)

**Pin Table for DP83956**  
(Configured as Port 1 Full AUI, Ports 2-3, AUI and Ports 4-7 Twisted-Pair)

| Pin Name        | Pin No. | Pin Name        | Pin No. | Pin Name        | Pin No. | Pin Name        | Pin No. | Pin Name        | Pin No. |
|-----------------|---------|-----------------|---------|-----------------|---------|-----------------|---------|-----------------|---------|
| GND             | 1       | V <sub>CC</sub> | 21      | RXM             | 41      | V <sub>CC</sub> | 61      | V <sub>CC</sub> | 81      |
| NC              | 2       | PKEN            | 22      | IRD             | 42      | RX2-            | 62      | CLK             | 82      |
| V <sub>CC</sub> | 3       | IRE             | 23      | IRC             | 43      | RX2+            | 63      | MLOAD           | 83      |
| RXI6+           | 4       | ACTNS           | 24      | STR             | 44      | CD2+            | 64      | WR              | 84      |
| RXI6-           | 5       | ACTND           | 25      | DFS             | 45      | CD2-            | 65      | RD              | 85      |
| TXO6P+          | 6       | ANYXNS          | 26      | BUFEN           | 46      | TX2+            | 66      | ACKI            | 86      |
| TXO6-           | 7       | ANYXND          | 27      | ACKO            | 47      | TX2-            | 67      | RXI4+           | 87      |
| TXO6+           | 8       | COLN            | 28      | CD1+            | 48      | NC              | 68      | RXI4-           | 88      |
| TXO6P-          | 9       | NC              | 29      | CD1-            | 49      | GND             | 69      | TXO4P+          | 89      |
| NC              | 10      | NC              | 30      | NC              | 50      | V <sub>CC</sub> | 70      | TXO4-           | 90      |
| GND             | 11      | D7              | 31      | RX1+            | 51      | TX3-            | 71      | TXO4+           | 91      |
| V <sub>CC</sub> | 12      | D6              | 32      | RX1-            | 52      | TX3+            | 72      | TXO4P-          | 92      |
| TXO7P-          | 13      | D5              | 33      | NC              | 53      | CD3-            | 73      | GND             | 93      |
| TXO7+           | 14      | D4              | 34      | V <sub>CC</sub> | 54      | CD3+            | 74      | V <sub>CC</sub> | 94      |
| TXO7-           | 15      | D3              | 35      | NC              | 55      | RX3-            | 75      | TXO5P-          | 95      |
| TXO7P+          | 16      | D2              | 36      | GND             | 56      | RX3+            | 76      | TXO5+           | 96      |
| RXI7+           | 17      | D1              | 37      | TX1+            | 57      | GND             | 77      | TXO5-           | 97      |
| RXI7-           | 18      | D0              | 38      | TX1-            | 58      | NC              | 78      | TXO5P+          | 98      |
| NC              | 19      | V <sub>CC</sub> | 39      | NC              | 59      | NC              | 79      | RXI5+           | 99      |
| GND             | 20      | GND             | 40      | GND             | 60      | NC              | 80      | RXI5-           | 100     |

Note: DP83956 will change from VLY package to VLJ package approximately Q3, 1993.



Top View

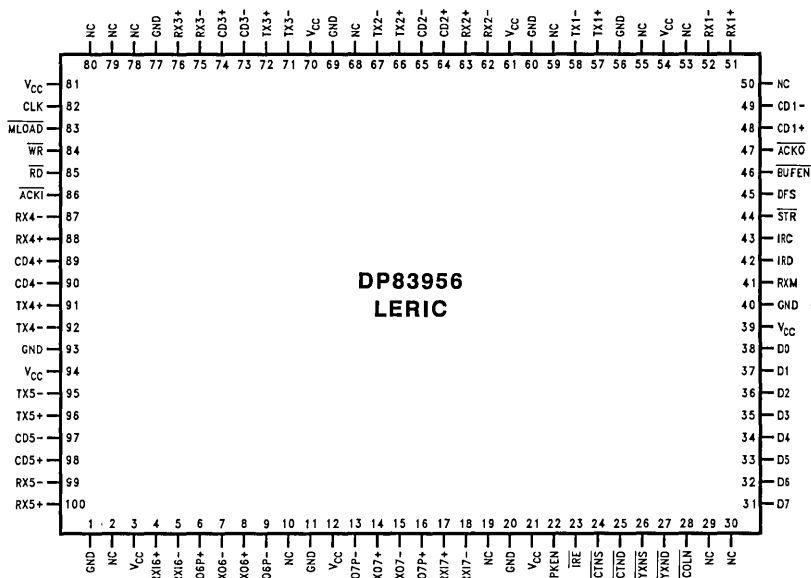
TL/F/11240-37

## 2.0 Connection Diagrams (Continued)

**Pin Table for DP83956**  
(Configured as Port 1 Full AUI, Ports 2-5, AUI, and Ports 6-7 Twisted-Pair)

| Pin Name        | Pin No. | Pin Name                   | Pin No. | Pin Name                  | Pin No. | Pin Name        | Pin No. | Pin Name                  | Pin No. |
|-----------------|---------|----------------------------|---------|---------------------------|---------|-----------------|---------|---------------------------|---------|
| GND             | 1       | V <sub>CC</sub>            | 21      | RXM                       | 41      | V <sub>CC</sub> | 61      | V <sub>CC</sub>           | 81      |
| NC              | 2       | PKEN                       | 22      | IRD                       | 42      | RX2-            | 62      | CLK                       | 82      |
| V <sub>CC</sub> | 3       | $\overline{\text{IRE}}$    | 23      | IRC                       | 43      | RX2+            | 63      | $\overline{\text{MLOAD}}$ | 83      |
| RXI6+           | 4       | $\overline{\text{ACTNS}}$  | 24      | $\overline{\text{STR}}$   | 44      | CD2+            | 64      | $\overline{\text{WR}}$    | 84      |
| RXI6-           | 5       | $\overline{\text{ACTND}}$  | 25      | DFS                       | 45      | CD2-            | 65      | $\overline{\text{RD}}$    | 85      |
| TXO6P+          | 6       | $\overline{\text{ANYXNS}}$ | 26      | $\overline{\text{BUFEN}}$ | 46      | TX2+            | 66      | $\overline{\text{ACKI}}$  | 86      |
| TXO6-           | 7       | $\overline{\text{ANYXND}}$ | 27      | $\overline{\text{ACKO}}$  | 47      | TX2-            | 67      | RX4-                      | 87      |
| TXO6+           | 8       | $\overline{\text{COLN}}$   | 28      | CD1+                      | 48      | NC              | 68      | RX4+                      | 88      |
| TXO6P-          | 9       | NC                         | 29      | CD1-                      | 49      | GND             | 69      | CD4+                      | 89      |
| NC              | 10      | NC                         | 30      | NC                        | 50      | V <sub>CC</sub> | 70      | CD4-                      | 90      |
| GND             | 11      | D7                         | 31      | RX1+                      | 51      | TX3-            | 71      | TX4+                      | 91      |
| V <sub>CC</sub> | 12      | D6                         | 32      | RX1-                      | 52      | TX3+            | 72      | TX4-                      | 92      |
| TXO7P-          | 13      | D5                         | 33      | NC                        | 53      | CD3-            | 73      | GND                       | 93      |
| TXO7+           | 14      | D4                         | 34      | V <sub>CC</sub>           | 54      | CD3+            | 74      | V <sub>CC</sub>           | 94      |
| TXO7-           | 15      | D3                         | 35      | NC                        | 55      | RX3+            | 75      | TX5-                      | 95      |
| TXO7P+          | 16      | D2                         | 36      | GND                       | 56      | RX3-            | 76      | TX5+                      | 96      |
| RXI7+           | 17      | D1                         | 37      | TX1+                      | 57      | GND             | 77      | CD5-                      | 97      |
| RXI7-           | 18      | D0                         | 38      | TX1-                      | 58      | NC              | 78      | CD5+                      | 98      |
| NC              | 19      | V <sub>CC</sub>            | 39      | NC                        | 59      | NC              | 79      | RX5-                      | 99      |
| GND             | 20      | GND                        | 40      | GND                       | 60      | NC              | 80      | RX5+                      | 100     |

Note: DP83956 will change from VLY package to VLJ package approximately Q3, 1993.



**DP83956**  
**LERIC**

Top View

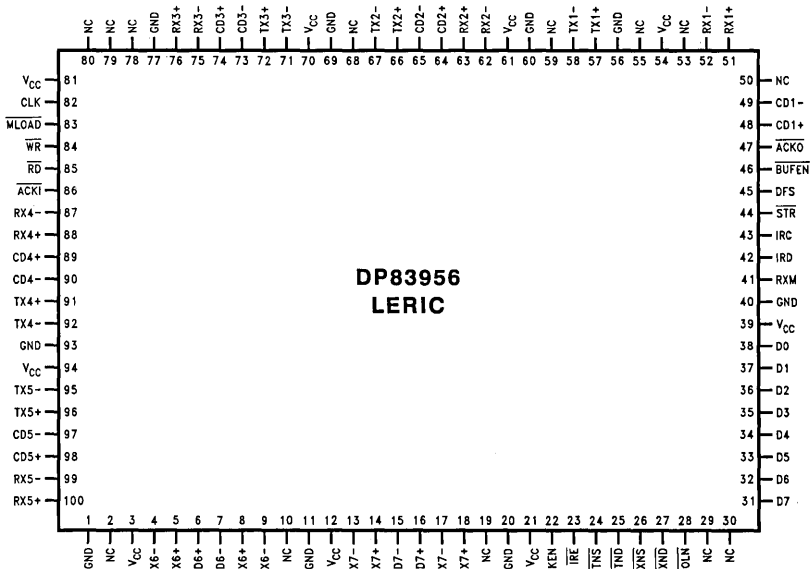
TL/F/11240-38

## 2.0 Connection Diagrams (Continued)

**Pin Table for DP83956**  
(Configured as Port 1 Full AUI, Ports 2-7 AUI)

| Pin Name        | Pin No. | Pin Name        | Pin No. | Pin Name        | Pin No. | Pin Name        | Pin No. | Pin Name        | Pin No. |
|-----------------|---------|-----------------|---------|-----------------|---------|-----------------|---------|-----------------|---------|
| GND             | 1       | V <sub>CC</sub> | 21      | RXM             | 41      | V <sub>CC</sub> | 61      | V <sub>CC</sub> | 81      |
| NC              | 2       | PKEN            | 22      | IRD             | 42      | RX2-            | 62      | CLK             | 82      |
| V <sub>CC</sub> | 3       | IRE             | 23      | IRC             | 43      | RX2+            | 63      | MLOAD           | 83      |
| RX6-            | 4       | ACTNS           | 24      | STR             | 44      | CD2+            | 64      | WR              | 84      |
| RX6+            | 5       | ACTND           | 25      | DFS             | 45      | CD2-            | 65      | RD              | 85      |
| CD6+            | 6       | ANYXNS          | 26      | BUFEN           | 46      | TX2+            | 66      | ACKI            | 86      |
| CD6-            | 7       | ANYXND          | 27      | ACKO            | 47      | TX2-            | 67      | RX4-            | 87      |
| TX6+            | 8       | COLN            | 28      | CD1+            | 48      | NC              | 68      | RX4+            | 88      |
| TX6-            | 9       | NC              | 29      | CD1-            | 49      | GND             | 69      | CD4+            | 89      |
| NC              | 10      | NC              | 30      | NC              | 50      | V <sub>CC</sub> | 70      | CD4-            | 90      |
| GND             | 11      | D7              | 31      | RX1+            | 51      | TX3-            | 71      | TX4+            | 91      |
| V <sub>CC</sub> | 12      | D6              | 32      | RX1-            | 52      | TX3+            | 72      | TX4-            | 92      |
| TX7-            | 13      | D5              | 33      | NC              | 53      | CD3-            | 73      | GND             | 93      |
| TX7+            | 14      | D4              | 34      | V <sub>CC</sub> | 54      | CD3+            | 74      | V <sub>CC</sub> | 94      |
| CD7-            | 15      | D3              | 35      | NC              | 55      | RX3+            | 75      | TX5-            | 95      |
| CD7+            | 16      | D2              | 36      | GND             | 56      | RX3-            | 76      | TX5+            | 96      |
| RX7-            | 17      | D1              | 37      | TX1+            | 57      | GND             | 77      | CD5-            | 97      |
| RX7+            | 18      | D0              | 38      | TX1-            | 58      | NC              | 78      | CD5+            | 98      |
| NC              | 19      | V <sub>CC</sub> | 39      | NC              | 59      | NC              | 79      | RX5+            | 99      |
| GND             | 20      | GND             | 40      | GND             | 60      | NC              | 80      | RX5-            | 100     |

Note: DP83956 will change from VLY package to VLJ package approximately Q3, 1993.



**DP83956**  
**LERIC**

Top View

### 3.0 Pin Description

| Pin Name                                                                                          | Driver Type | I/O | Description                                        |
|---------------------------------------------------------------------------------------------------|-------------|-----|----------------------------------------------------|
| <b>NETWORK INTERFACE PINS (On-Chip Transceiver Mode)</b>                                          |             |     |                                                    |
| RXI2- to RXI7-                                                                                    | TP          | I   | Twisted-Pair Receive Input Negative                |
| RXI2+ to RXI7+                                                                                    | TP          | I   | Twisted-Pair Receive Input Positive                |
| TXOP2- to TXOP7-                                                                                  | TT          | O   | Twisted-Pair Pre-Emphasis Transmit Output Negative |
| TXO2- to TXO7-                                                                                    | TT          | O   | Twisted-Pair Transmit Output Negative              |
| TXO2+ to TXO7+                                                                                    | TT          | O   | Twisted-Pair Transmit Output Positive              |
| TXOP2+ to TXOP7+                                                                                  | TT          | O   | Twisted-Pair Pre-Emphasis Transmit Output Positive |
| CD1+                                                                                              | AL          | I   | AUI Collision Detect Input Positive                |
| CD1-                                                                                              | AL          | I   | AUI Collision Detect Input Negative                |
| RX1+                                                                                              | AL          | I   | AUI Receive Input Positive                         |
| RX1-                                                                                              | AL          | I   | AUI Receive Input Negative                         |
| TX1+                                                                                              | AD          | O   | AUI Transmit Output Positive                       |
| TX1-                                                                                              | AD          | O   | AUI Transmit Output Negative                       |
| <b>NETWORK INTERFACE PINS (External Transceiver Mode AUI Signal Level Compatibility Selected)</b> |             |     |                                                    |
| TX2+ to TX7+                                                                                      | AL          | O   | Transmit Output Positive                           |
| TX2- to TX7-                                                                                      | AL          | O   | Transmit Output Negative                           |
| CD2+ to CD7+                                                                                      | AL          | I   | Collision Input Positive                           |
| CD2- to CD7-                                                                                      | AL          | I   | Collision Input Negative                           |
| RX2+ to RX7+                                                                                      | AL          | I   | Receive Input Positive                             |
| RX2- to RX7-                                                                                      | AL          | I   | Receive Input Negative                             |
| CD1+                                                                                              | AL          | I   | AUI Collision Detect Input Positive                |
| CD1-                                                                                              | AL          | I   | AUI Collision Detect Input Negative                |
| RX1+                                                                                              | AL          | I   | AUI Receive Input Positive                         |
| RX1-                                                                                              | AL          | I   | AUI Receive Input Negative                         |
| TX1+                                                                                              | AD          | O   | AUI Transmit Output Positive                       |
| TX1-                                                                                              | AD          | O   | AUI Transmit Output Negative                       |

**Note:** AD = AUI level and Drive compatible  
 TP = Twisted-Pair interface compatible  
 AL = AUI Level compatible  
 TT = TTL compatible  
 I = Input  
 O = Output

### 3.0 Pin Description (Continued)

| Pin Name                    | Driver Type | I/O  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-----------------------------|-------------|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>PROCESSOR BUS PINS</b>   |             |      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| STR                         | C           | O    | Display Update <b>ST</b> robe: This signal controls the latching of display data for network ports into the off chip display latches.<br>During processor access cycles (read or write is asserted) this signal is inactive (high).                                                                                                                                                                                                                                                 |
| D(7:0)                      | TT          | B, Z | <b>Data Bus:</b><br><b>Display Update Cycles:</b> These pins become outputs providing display data and port address information.<br><b>Processor Access Cycles:</b> Address input D(7:4) and Data input or output D(3:0) is performed via these pins. The read, write and reset inputs control the direction of the signals.<br><b>Note:</b> The data pins remain in their display update function, (i.e., asserted as outputs) unless either the read or write strobe is asserted. |
| DFS                         | C           | O    | Display Frozen Strobe: The assertion of the DFS signal, active high, at the end of the transmission of each packet indicates that the status of that packet is frozen on the LEDs until the beginning of the next received packet or for a maximum of 30 ms.                                                                                                                                                                                                                        |
| BUFEN                       | C           | O    | <b>BU</b> ffer <b>EN</b> able: This output controls the TRI-STATE® operation of the bus transceiver which provides the interface between the LERIC's data pins and the processor's data bus.<br><b>Note:</b> The buffer enable output indicates the function of the data pins. When it is high they are performing display update cycles, when it is low a processor access or MLOAD cycle is occurring.                                                                            |
| WR                          | TT          | I    | <b>WR</b> ite Strobe: Strobe from the CPU used to write an internal register defined by the D(7:4) inputs.                                                                                                                                                                                                                                                                                                                                                                          |
| RD                          | TT          | I    | <b>RD</b> Strobe: Strobe from the CPU used to read an internal register defined by the D(7:4) inputs.                                                                                                                                                                                                                                                                                                                                                                               |
| MLOAD                       | TT          | I    | Device <b>MLOAD</b> and Reset: When this input is low all of the RIC's state machines and network ports are reset and held inactive. On the rising edge of MLOAD the logic levels present on the D(7:0) pins are latched into the LERIC's configuration registers. The rising edge of MLOAD also signals the beginning of the display test operation.                                                                                                                               |
| <b>INTER-LERIC BUS PINS</b> |             |      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| ACKI                        | TT          | I    | <b>ACK</b> nowledge <b>I</b> nter-LERIC: Input to the network ports' arbitration chain.                                                                                                                                                                                                                                                                                                                                                                                             |
| ACKO                        | TT          | O    | <b>ACK</b> nowledge <b>O</b> utput: Output from the network ports' arbitration chain.                                                                                                                                                                                                                                                                                                                                                                                               |
| IRD                         | TT          | B, Z | <b>Inter-LERIC Data:</b> When asserted as an output this signal provides a serial data stream in NRZ format. The signal is asserted by a LERIC when it is receiving data from one of its network segments. The default condition of this signal is to be an input. In this state it may be driven by other devices on the Inter-LERIC bus.                                                                                                                                          |

### 3.0 Pin Description (Continued)

| Pin Name                                | Driver Type | I/O  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|-----------------------------------------|-------------|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>INTER-LERIC BUS PINS (Continued)</b> |             |      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| IRE                                     | TT          | B, Z | <b>Inter-LERIC Enable:</b> When asserted as an output this signal provides an activity framing enable for the serial data stream. The signal is asserted by a LERIC when it is receiving data from one of its network segments. The default condition of this signal is to be an input. In this state it may be driven by other devices on the Inter-LERIC bus.                                                                                                           |
| IRC                                     | TT          | B, Z | <b>Inter-LERIC Clock:</b> When asserted as an output this signal provides a clock signal for the serial data stream. Data (IRD) is changed on the falling edge of the clock. The signal is asserted by a LERIC when it is receiving data from one of its network segments. The default condition of this signal is to be an input. When an input, IRD is sampled on the rising edge of the clock. In this state it may be driven by other devices on the Inter-LERIC bus. |
| COLN                                    | TT          | B, Z | <b>COL</b> lision on Port N: This denotes that a collision is occurring on the port receiving the data packet (Port N). The default condition of this signal is to be an input. In this state it may be driven by other devices on the Inter-LERIC bus.                                                                                                                                                                                                                   |
| CLK                                     | TT          | I    | <b>20 MHz Clock Input:</b> This input is used to generate the LERIC's timing reference for the state machines, and phase lock loop decoder. The 20 MHz clock should have a 0.01% frequency tolerance and 40%–60% duty cycle or better (i.e. 50/50 duty cycle).                                                                                                                                                                                                            |
| <b>POWER AND GROUND PINS</b>            |             |      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| VCC                                     |             |      | Positive Supply                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| GND                                     |             |      | Negative Supply                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>EXTERNAL DECODER PINS</b>            |             |      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| RXM                                     | TT          | O    | <b>Receive Data Manchester Format:</b> This output makes the data, in Manchester format, received by port N available for test purposes. If not used for testing, this pin should be left open.                                                                                                                                                                                                                                                                           |

**Note:** TT = TTL compatible  
 B = Bi-directional  
 C = CMOS compatible  
 OD = Open Drain  
 I = Input  
 O = Output  
 Z = TRI-STATE

### 3.0 Pin Description (Continued)

#### Pin Description for DP83955

| Pin No. | Pin Name | Driver Type | I/O | Description                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------|----------|-------------|-----|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 30      | ACTN     | OD          | B   | <b>ACTivity on Port N:</b> This is a bidirectional signal. The LERIC asserts this signal when data or collision information is received from one of its network segments. The LERIC senses this signal when this LERIC or another LERIC in a multi-LERIC system is receiving data or collision information.                                                                                                            |
| 31      | ANYXN    | OD          | B   | <b>Activity on ANY Port EXcluding Port N:</b> This is a bidirectional signal. The LERIC asserts this signal when a transmit collision is experienced or multiple ports have active collisions on their network segments. The LERIC senses this signal when this LERIC or other LERICs in a multi-LERIC system are experiencing transmit collisions or multiple ports have active collisions on their network segments. |

B = Bi-directional

#### Pin Description for DP83956

| Pin No. | Pin Name | Driver Type | I/O | Description                                                                                                                                                                                                                                                                                                                                               |
|---------|----------|-------------|-----|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 25      | ACTND    | OD          | O   | <b>ACTivity on Port N Drive:</b> The LERIC asserts this signal when data or collision information is received from one of its network segments.                                                                                                                                                                                                           |
| 24      | ACTNS    | TT          | I   | <b>ACTivity on Port N Sense:</b> The LERIC senses this signal when this LERIC or another LERIC in a multi-LERIC system is receiving data or collision information.                                                                                                                                                                                        |
| 27      | ANYXND   | OD          | O   | <b>Activity on ANY Port EXcluding Port N Drive:</b> The LERIC asserts this signal when a transmit collision is experienced or multiple ports have active collisions on their network segments.                                                                                                                                                            |
| 26      | ANYXNS   | TT          | I   | <b>Activity on ANY Port EXcluding Port N Sense:</b> The LERIC senses this signal when this LERIC or other LERICs in a multi-LERIC system are experiencing transmit collisions or multiple ports have active collisions on their network segments.                                                                                                         |
| 22      | PKEN     | C           | O   | <b>PacKet ENable:</b> This signal acts as an active high enable for an external bus transceiver (if required) for the IRE, IRC, IRD, and COLN signals. When high, the bus transceiver should be transmitting on to the bus, i.e., this LERIC is driving the IRD, IRE, IRC, and COLN bus lines. When low, the bus transceiver should receive from the bus. |

TT = TTL compatible, C = CMOS compatible, OD = Open Drain, I = Input, O = Output



4.0 Block Diagram  
4.0.1 DP83955 BLOCK DIAGRAM

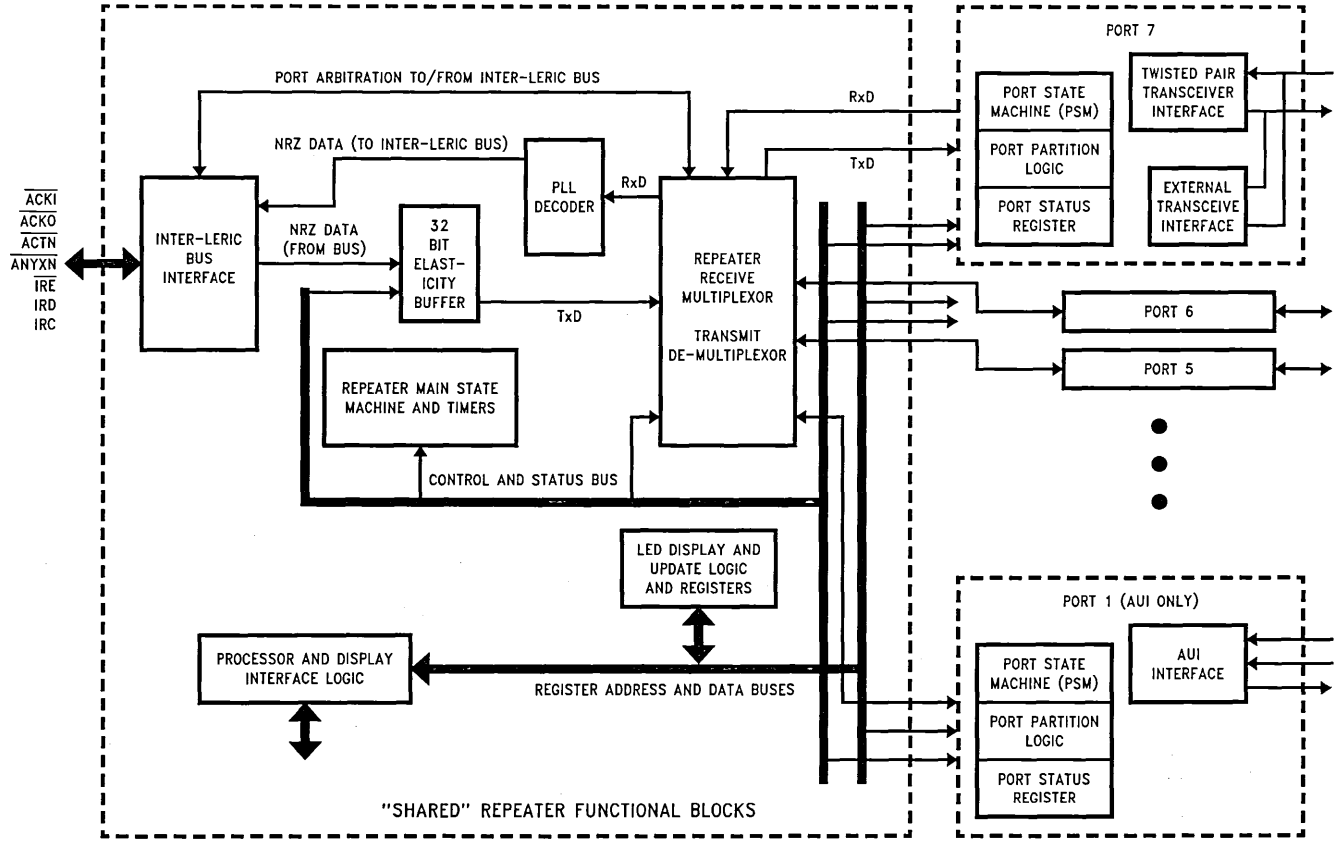


FIGURE 4-1a. LERIC Block Diagram for DP83955

TL/F/11240-6

3-185

4.0 Block Diagram (Continued)  
4.0.2 DP83956 BLOCK DIAGRAM

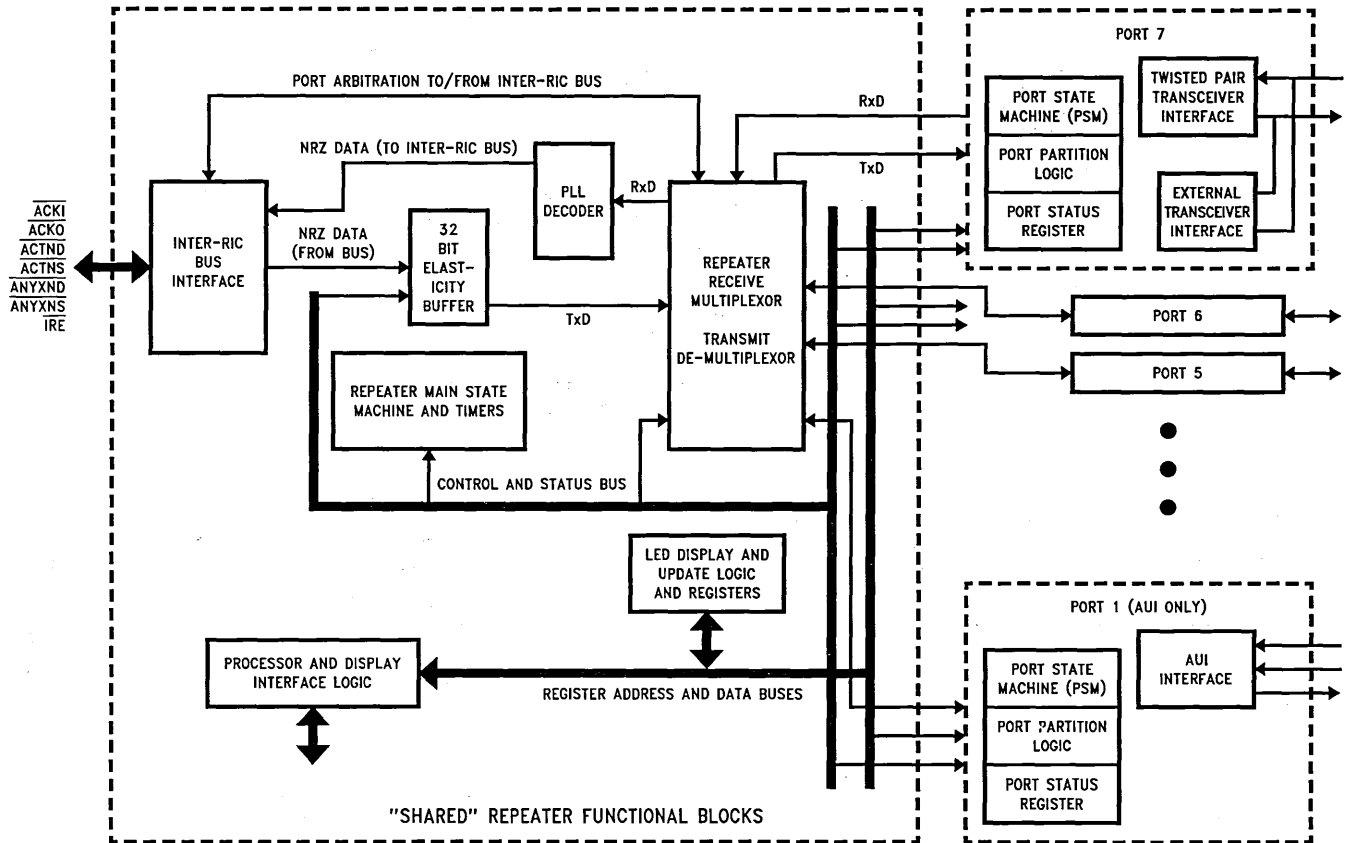


FIGURE 4-1b. LERIC Block Diagram for DP83956

TL/F/11240-40

## 5.0 Functional Description

The IEEE 802.3 repeater specification details a number of functions a repeater system must perform. These requirements allied with a need for the implementation to be multiport strongly favors the choice of a modular design style. In such a design, functionality is split between those tasks common to all data channels and those exclusive to each individual channel. The LERIC, much like the DP83950 RIC, follows this approach. Certain functional blocks are replicated for each network attachment (also known as a repeater port), and others are shared. The following section briefly describes the functional blocks in the LERIC.

### 5.1 OVERVIEW OF LERIC FUNCTIONS

#### Segment Specific Block: Network Port

As shown in the Block Diagram, the segment specific blocks consist of:

1. One or more physical layer interfaces.
2. A logic block required for performing repeater operations upon that particular segment. This is known as the "port" logic since it is the access "port" the segment has to the rest of the network.

This function is repeated 7 times in the LERIC (one for each port) and is shown on the right side of the Block Diagram, *Figure 4-1*.

The physical layer interfaces provided depends upon the port under examination. Port 1 has an AUI compliant interface for use with AUI compatible transceiver boxes and cable. Ports 2 to 7 may be configured for use with one of two interfaces: twisted pair or an external transceiver. The former utilizes the LERIC's on-chip 10BASE-T transceivers, the latter allows connection to external transceivers. When using the external transceiver mode the interface is AUI compatible. Although AUI compatible transceivers are supported the interface is not designed for use with an interface cable, thus the transceivers are necessarily internal to the repeater equipment.

Inside the port logic there are 3 distinct functions:

1. The port state machine (PSM) is required to perform data and collision repetition as described by the repeater specification, for example, it determines whether this port should be receiving from or transmitting to its network segment.
2. The port partition logic implements the segment partitioning algorithm. This algorithm is defined by the IEEE specification and is used to protect the network from malfunctioning segments.
3. The port status register reflects the current status of the port. It may be accessed by a system processor to obtain this status or to perform certain port configuration operations, such as port disable and squelch level selection.

#### Shared Functional Blocks: Repeater Core Logic

The shared functional blocks consist of the Repeater Main Status Machine (MSM) and Timers, a 32-bit Elasticity Buffer, PLL Decoder, and Receive and Transmit Multiplexers. These blocks perform the majority of the operations needed to fulfill the requirements of the IEEE repeater specification.

When a packet is received by a port, it is sent via the Receive Multiplexer to the PLL Decoder. Notification of the data and collision status is sent to the main state machine via the receive multiplexer and collision activity status signals. This enables the main state machine to determine the source of the data to be repeated and the type of data to be

transmitted. The transmit data may be either the received packet's data field or a preamble/jam pattern consisting of a 1010 . . . bit pattern.

Associated with the main state machine are a series of timers. These ensure various IEEE specification times (referred to as the TW1 to TW6 times) are fulfilled.

A repeater unit is required to meet the same signal jitter performance as any receiving node attached to a network segment. Consequently, a phase locked loop Manchester decoder is required so that the packet may be decoded, and the jitter accumulated over the receiving segment eliminated. The decode logic outputs data in NRZ format with an associated clock and enable. In this form the packet is in a convenient format for transfer to other devices, such as network controllers and other LERICs, via the Inter-LERIC bus (described later). The data may then be re-encoded into Manchester data and transmitted.

Reception and transmission via physical layer transceiver units causes a loss of bits in the preamble field of a data packet. The repeater specification requires this loss to be compensated for. To accomplish this an elasticity buffer is employed to temporarily store bits in the data field of the packet.

The sequence of operation is as follows. Soon after the network segment receiving the data packet has been identified, the LERIC begins to transmit the packet preamble pattern (1010 . . .) onto the other network segments. While the preamble is being transmitted the Elasticity Buffer monitors the decoded received clock and data signals (this is done via the Inter-LERIC/Inter-RIC bus as described later). When the start of frame delimiter "SFD" is detected the received data stream is written into the elasticity buffer. Removal of data from the buffer for retransmission is not allowed until a valid length preamble pattern has been transmitted.

#### Inter-LERIC/Inter-RIC Bus Interface

The LERIC can be cascaded either to other LERICs or RICs to facilitate the design of large multiport repeaters. The split of functions already described allows data packets and collision status to be transferred between multiple LERICs, and at the same time the multiple LERICs still behave as a single logical repeater. Since all LERICs in a repeater system are identical and capable of performing any of the repetition operations, the failure of one LERIC will not cause the failure of the entire system. This is an important issue in large multiport repeaters.

DP83955's communicate via a specialized interface known as the Inter-LERIC bus. DP83956s can communicate with other DP83956s and/or DP83950s via the Inter-RIC bus. These allow the data packets to be transferred from the receiving LERIC to the other LERICs in the system. These LERICs then transmit the data stream to their segments. Just as important as data transfer is the notification of collisions occurring across the network. The Inter-LERIC/Inter-RIC bus has a set of status lines capable of conveying collision information between LERICs to ensure their main state machines operate in the appropriate manner.

#### LED Interface

Repeater systems usually possess optical displays indicating network activity and the status of specific repeater operations. The LERIC's display update block provides the system designer with a wide variety of indicators. The display

## 5.0 Functional Description (Continued)

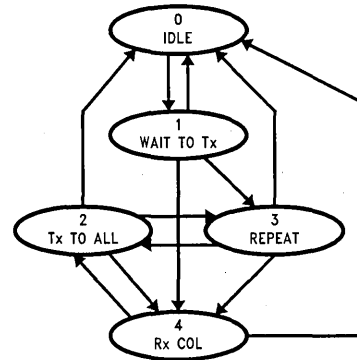
updates are completely autonomous and merely require SSI logic devices to drive the display devices, usually made up of light emitting diodes, LEDs. The status display is very flexible, allowing the user to choose those indicators appropriate for the specification of the equipment. The Display Frozen Strobe ( $\overline{DFS}$ ) may be used to latch the various indicators which are frozen at the end of the activity. The LED display will be frozen for 30 ms after the end of the activity, or until a new activity has started, whichever is shorter. Note that the complete LED display cycle for all the ports takes approximately 1.6  $\mu$ s.

### Processor Interface

The LERIC's processor interface allows connection to a system processor (or a simple read/write logic interface). Data transfer occurs via a 4-bit bidirectional data bus, and 4-bit address bus. Display update cycles and processor accesses occur utilizing the same bus. An on-chip arbiter in the processor/display block schedules and controls the accesses and ensures the correct information is written into the display latches. During the display update cycles the LERIC behaves as a master of its bus. This is the default state of the bus. Consequently, a TRI-STATE buffer must be placed between the LERIC and the system processor's data bus. This ensures bus contention is avoided during simultaneous display update cycles and processor accesses of other devices on the system bus. When the processor accesses a LERIC register, the LERIC enables the data buffer and selects the operation, either input to or output from the data pins.

### 5.2 DESCRIPTION OF REPEATER OPERATIONS

In order to implement a multi-chip repeater system which behaves as though it were a single logical repeater, special consideration must be paid to the data path used in packet repetition. For example, where in the path are specific operations such as Manchester decoding and elasticity buffering performed. Also the system's state machines which utilize available network activity signals, must be able to accommodate the various packet repetition and collision scenarios detailed in the IEEE 802.3 repeater specification.



TL/F/11240-7

**FIGURE 5-1. Inter-LERIC/Inter-RIC Bus State Diagram**

The LERIC contains two types of interacting state machines. These are:

1. Port State Machines (PSMs). Every network attachment has its own PSM.
2. Main State Machine (MSM). This state machine controls the shared functional blocks as shown in the block diagram *Figure 4-1*.

### Repeater Port and Main State Machines

These two state machines are described in the following sections. Reference is made to expressions used in the IEEE 802.3 Repeater specification. For the precise definition of these terms please refer to the IEEE specifications. To avoid confusion with the LERIC's implementation, where references are made to repeater states or terms as described in the IEEE specification, these items are written in *italics*. The IEEE state diagram is shown in *Figure 5-2*, the Inter-LERIC/Inter-RIC bus state diagram is shown in *Figure 5-1*.

5.0 Functional Description (Continued)

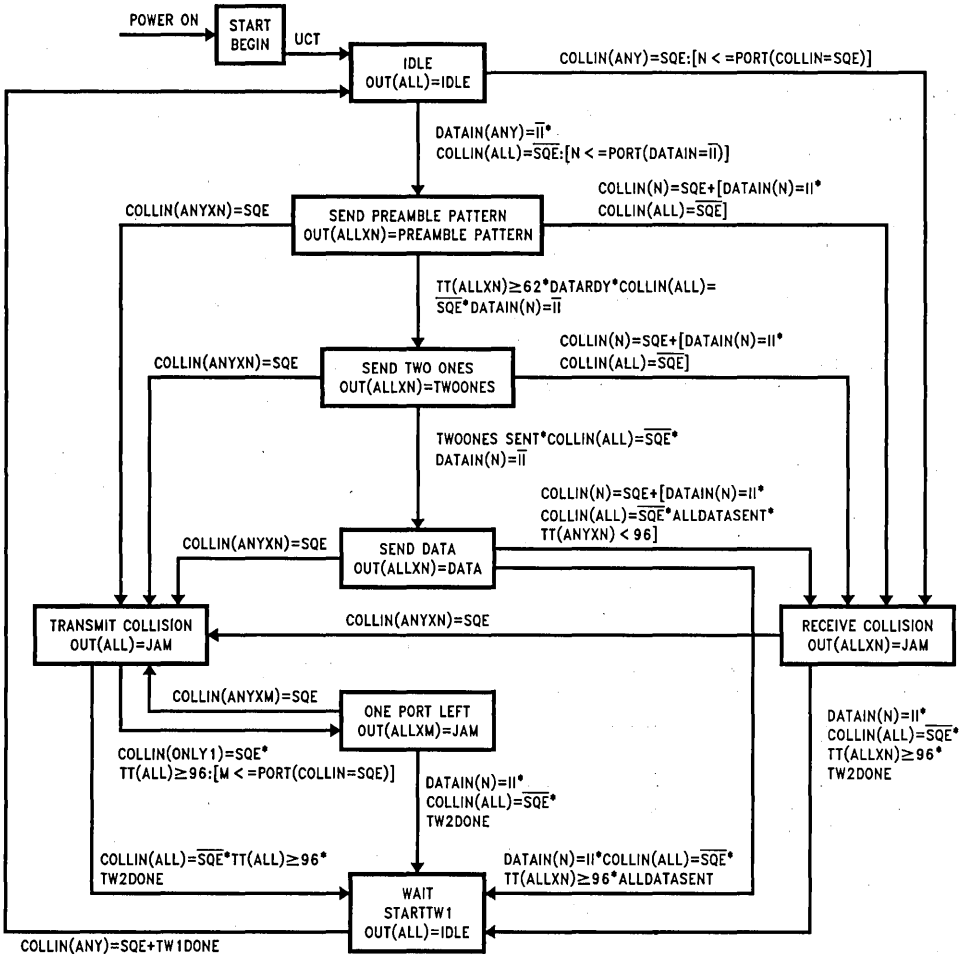


FIGURE 5-2. IEEE Repeater Main State Diagram

TL/F/11240-8

## 5.0 Functional Description (Continued)

### Port State Machine (PSM)

There are two primary functions for the PSM as follows:

1. Control the transmission of repeated data and jam signals over the attached segment.
2. Decide whether a port will be the source of data or collision information which will be repeated over the network. This repeater port is known as *PORT N*. An arbitration process is required to enable the repeater to transition from the *IDLE* state to the *SEND PREAMBLE PATTERN* or *RECEIVE COLLISION* states, see *Figure 5-2*. This process is used to locate the port which will be *PORT N* for that particular packet. The data received from this port is directed to the PLL decoder and transmitted over the Inter-LERIC bus. If the repeater enters the *TRANSMIT COLLISION* state a further arbitration operation is performed to determine which port is *PORT M*. *PORT M* is differentiated from the repeater's other ports if the repeater enters the *ONE PORT LEFT* state. In this state *PORT M* does not transmit to its segment; where as all other ports are still required to transmit to their segments.

### Main State Machine (MSM)

The MSM controls the operation of the shared functional blocks in each LERIC as shown in the block diagram, *Figure 4-7*, and it performs the majority of the data and collision propagation operations as defined by the IEEE specification, these include those shown in *Table 5-1*.

The interaction of the main and port state machines is visible, in part, by observing the Inter-LERIC bus.

TABLE 5-1. Main State Machine Operations

| Function                          | Action                                                                                                                                                                                 |
|-----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Preamble Regeneration             | Restore the length of the preamble pattern to the defined size.                                                                                                                        |
| Fragment Extension                | Extend received data or collision fragments to meet the minimum fragment length of 96 bits.                                                                                            |
| Elasticity Buffer Control         | A portion of the received packet may require storage in an Elasticity Buffer to accommodate preamble regeneration.                                                                     |
| Jam/Preamble Pattern Generation   | In cases of receive or transmit collisions a LERIC is required to transmit a jam pattern (1010 ... ).<br><b>Note:</b> This pattern is the same as that used for preamble regeneration. |
| Transmit Collision Enforcement    | Once the <i>TRANSMIT COLLISION</i> state is entered a repeater is required to stay in this state for at least 96 network bit times.                                                    |
| Data Encoding Control             | NRZ format data from the elasticity buffer must be encoded into Manchester format data prior to retransmission.                                                                        |
| <i>T<sub>w1</sub></i> Enforcement | Enforce the Transmit Recovery Time specification.                                                                                                                                      |
| <i>T<sub>w2</sub></i> Enforcement | Enforce Carrier Recovery Time specification on all ports with active collisions.                                                                                                       |

### Inter-LERIC Bus Operation

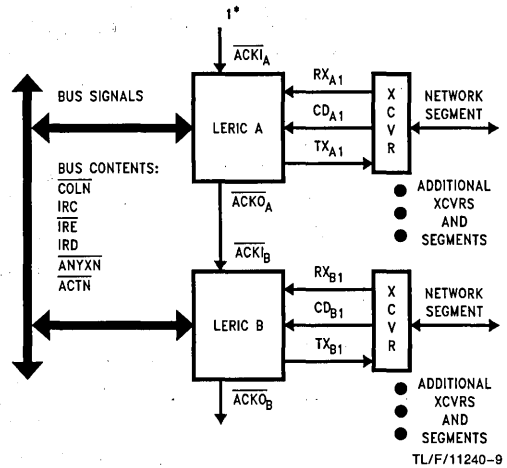
#### Overview

The Inter-LERIC Bus, like the Inter-RIC Bus, consists of eight signals. These signals implement a protocol which may be used to connect multiple LERICs together. In this configuration, the logical function of a single repeater is maintained. The resulting multi-LERIC system is compliant to the IEEE 802.3 Repeater Specification and may connect several hundred network segments. An example of a multi-LERIC system is shown in *Figure 5-3*.

The Inter-LERIC Bus connects multiple LERICs to realize the following operations:

- Port N* Identification (which port the repeater receives data from)
- Port M* Identification (which port is the last one experiencing a collision)
- Data Transfer
- RECEIVE COLLISION* identification
- TRANSMIT COLLISION* identification
- DISABLE OUTPUT* (jabber protection)

The following tables briefly describe the operation of each bus signal, the conditions required for a LERIC to assert a signal and which LERICs (in a multi-LERIC system) would monitor a signal:



\*Note 1: This input is tied at a logic high state.

FIGURE 5-3. LERIC System Topology

## 5.0 Functional Description (Continued)

### ACKI

|                                                             |                                                                                                                                                                                                                                                                                 |
|-------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Function</b>                                             | Input signal to the PSM arbitration chain. This chain is employed to identify <i>PORT N</i> and <i>PORT M</i> .<br><b>Note:</b> A LERIC which contains <i>PORT N</i> or <i>PORT M</i> may be identified by its <i>ACKO</i> signal being low when its <i>ACKI</i> input is high. |
| <b>Conditions required for a LERIC to drive this signal</b> | Not Applicable                                                                                                                                                                                                                                                                  |
| <b>LERIC Receiving the Signal</b>                           | This is dependent upon the method used to cascade LERICs, described in Section 5.3.                                                                                                                                                                                             |

### ACKO

|                                                             |                                                                                     |
|-------------------------------------------------------------|-------------------------------------------------------------------------------------|
| <b>Function</b>                                             | Output signal from the PSM arbitration chain.                                       |
| <b>Conditions required for a LERIC to drive this signal</b> | This is dependent upon the method used to cascade LERICs, described in Section 5.3. |
| <b>LERIC Receiving the Signal</b>                           | Not Applicable                                                                      |

### ACTN

|                                                             |                                                                                                                                                                                             |
|-------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Function</b>                                             | This signal denotes there is activity on <i>PORT N</i> or <i>PORT M</i> .                                                                                                                   |
| <b>Conditions required for a LERIC to drive this signal</b> | A LERIC must contain <i>PORT N</i> or <i>PORT M</i> .<br><b>Note:</b> Although this signal normally has only one source asserting the signal active it is used in a wired-OR configuration. |
| <b>LERIC Receiving the Signal</b>                           | The signal is monitored by all LERICs in the repeater system.                                                                                                                               |

### ANYXN

|                                                             |                                                                                                                   |
|-------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------|
| <b>Function</b>                                             | This signal denotes that a repeater port that is not <i>Port N</i> or <i>Port M</i> is experiencing a collision.  |
| <b>Conditions required for a LERIC to drive this signal</b> | Any LERIC which satisfies the above condition.<br><b>Note:</b> This bus line is used in a wired-OR configuration. |
| <b>LERIC Receiving the Signal</b>                           | The signal is monitored by all LERICs in the repeater system.                                                     |

### COLN

|                                                             |                                                                     |
|-------------------------------------------------------------|---------------------------------------------------------------------|
| <b>Function</b>                                             | Denotes <i>PORT N</i> or <i>PORT M</i> is experiencing a collision. |
| <b>Conditions required for a LERIC to drive this signal</b> | A LERIC must contain <i>PORT N</i> or <i>PORT M</i> .               |
| <b>LERIC Receiving the Signal</b>                           | The Signal is monitored by all other LERICs in the repeater system. |

### IRE

|                                                             |                                                                             |
|-------------------------------------------------------------|-----------------------------------------------------------------------------|
| <b>Function</b>                                             | This signal acts as an activity framing signal for the IRC and IRD signals. |
| <b>Conditions required for a LERIC to drive this signal</b> | A LERIC must contain <i>PORT N</i> .                                        |
| <b>LERIC Receiving the Signal</b>                           | The Signal is monitored by all other LERICs in the repeater system.         |

### IRD

|                                                             |                                                                                                   |
|-------------------------------------------------------------|---------------------------------------------------------------------------------------------------|
| <b>Function</b>                                             | Decoded serial data, in NRZ format, received from the network segment attached to <i>PORT N</i> . |
| <b>Conditions required for a LERIC to drive this signal</b> | A LERIC must contain <i>PORT N</i> .                                                              |
| <b>LERIC Receiving the Signal</b>                           | The signal is monitored by all other LERICs in the repeater system.                               |

### IRC

|                                                             |                                                                     |
|-------------------------------------------------------------|---------------------------------------------------------------------|
| <b>Function</b>                                             | Clock signal associated with IRD and IRE.                           |
| <b>Conditions required for a LERIC to drive this signal</b> | A LERIC must contain <i>PORT N</i> .                                |
| <b>LERIC Receiving the Signal</b>                           | The signal is monitored by all other LERICs in the repeater system. |

## 5.0 Functional Description (Continued)

### Methods of LERIC Cascading

In order to build multi-LERIC repeaters, *PORT N* and *PORT M* identification must be performed across all the LERICs in the system. Inside each LERIC the PSMs are arranged in a logical arbitration chain where Port 1 is the highest and Port 7 the lowest.

The top of the chain, the input to Port 1 is accessible to the user via the LERIC's  $\overline{ACKI}$  input pin. The output from the bottom of the chain becomes the  $\overline{ACKO}$  output pin. In a single LERIC system *PORT N* is defined as the highest port in the arbitration chain with receive or collision activity. *PORT N* identification is performed when the repeater is in the *IDLE* state. *PORT M* is defined as the highest port in the chain with a collision when the repeater leaves the *TRANSMIT COLLISION* state. In order for the arbitration chain to function, all that needs to be done is to tie the  $\overline{ACKI}$  signal to a logic high state. In multi-LERIC systems there are two methods to propagate the arbitration chain between LERICs:

The first and most straightforward way is to extend the arbitration chain by daisy-chaining the  $\overline{ACKI}$ – $\overline{ACKO}$  signals between LERICs. In this approach one LERIC is placed at the top of the chain (its  $\overline{ACKI}$  input is tied high), then the  $\overline{ACKO}$  signal from this LERIC is sent to the  $\overline{ACKI}$  input of the next LERIC and so on. This arrangement is simple to implement but it places some topological restrictions upon the repeater system. In particular, when the repeater is constructed using a backplane with removable printed circuit boards containing the LERICs, if one of the boards is removed then the  $\overline{ACKI}$ – $\overline{ACKO}$  chain will be broken and the repeater will not operate correctly.

The second method of *PORT N* or *M* identification avoids this problem. This second technique relies on an external parallel arbiter which monitors all of the LERICs'  $\overline{ACKO}$  signals and responds to the LERIC with the highest priority. In this scheme each LERIC is assigned with a priority level. One method of doing this is to assign a priority number which reflects the position of a LERIC board on the repeater backplane (i.e., its slot number). When a LERIC experiences receive activity and the repeater system is in the *IDLE* state, the LERIC board will assert  $\overline{ACKO}$ . External arbitration logic drives the identification number onto an arbitration bus and the LERIC containing *PORT N* will be identified. An identical procedure is used in the *TRANSMIT COLLISION* state to identify *PORT M*. This parallel means of arbitration is not subject to the problems caused by missing boards (i.e., empty slots in the backplane). The logic associated with asserting this arbitration vector in the various packet repetition scenarios could be implemented in PAL® or GAL® type devices.

To perform *PORT N* or *M* arbitration, both of the above methods employ the same signals:  $\overline{ACKI}$ ,  $\overline{ACKO}$ , and  $\overline{ACTN}$ .

The Inter-LERIC bus allows multi-LERIC operations to be performed in exactly the same manner as if there is only a single LERIC in the system. The simplest way to describe the operation of Inter-LERIC bus is to see how it is used in a number of common packet repetition scenarios. Throughout this description the LERICs are presumed to be operating in external transceiver mode. This is advantageous for the explanation since the receive, transmit and collision signals from each network segment are observable. In internal transceiver mode this is not the case, since the collision signal for the non-AUI ports is derived by the transceivers inside the LERIC.

### 5.3 EXAMPLES OF PACKET REPETITION SCENARIOS

#### Data Repetition

The simplest packet operation performed over the Inter-LERIC Bus is data repetition. In this operation a data packet is received at one port and transmitted to all other segments.

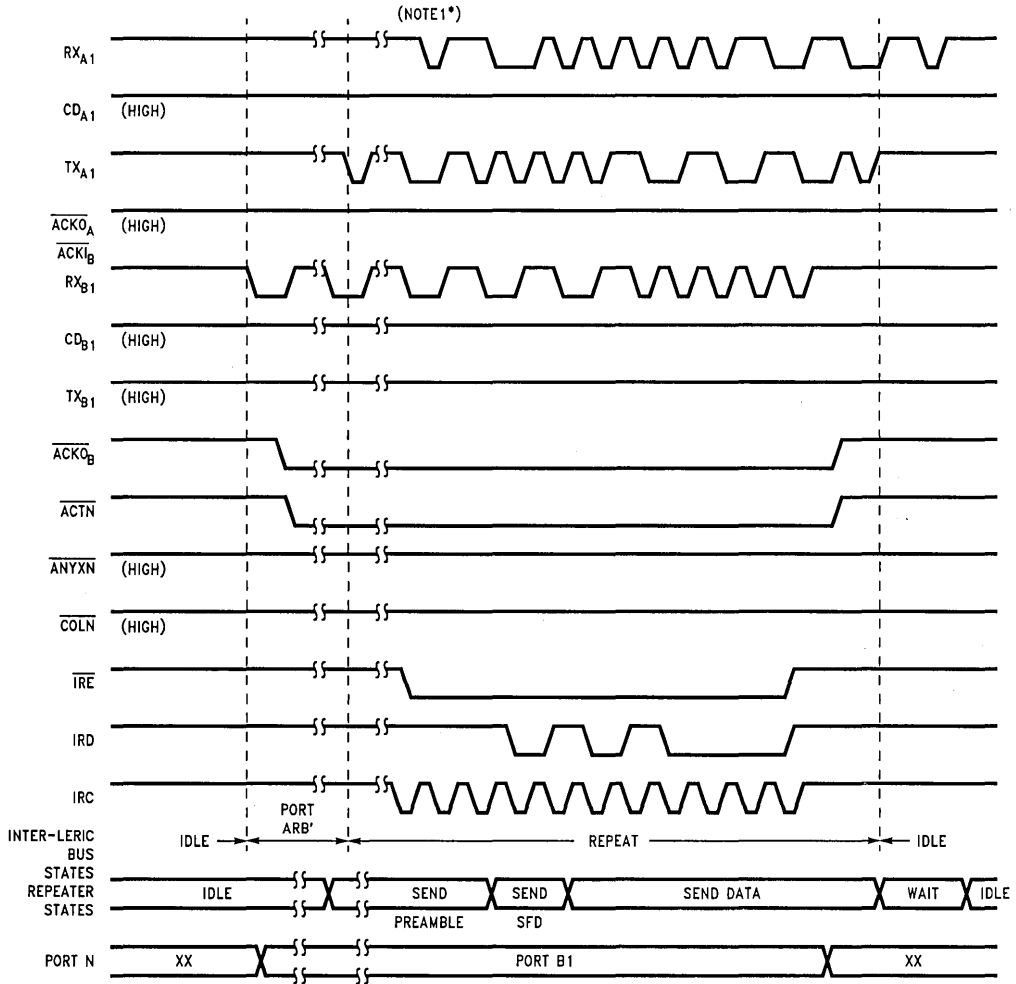
The first task to be performed is *PORT N* identification. This is an arbitration process performed by the Port State Machines in the system. In situations where two or more ports simultaneously receive packets the Inter-LERIC bus operates by choosing one of the active ports and forcing the others to transmit data. This is done to faithfully follow the IEEE specification's allowed exit paths from the *IDLE* state (i.e., to the *SEND PREAMBLE PATTERN* or *RECEIVE COLLISION* states).

The packet begins with a preamble pattern derived from the LERIC's on chip jam/preamble generator. The data received at *PORT N* is directed through the receive multiplexer to the PLL decoder. Once phase lock has been achieved, the decoded data, in NRZ format, with its associated clock and enable signals are asserted onto the IRD, IRE and IRC Inter-LERIC bus lines. This serial data stream is received from the bus by all LERICs in the repeater and directed to their Elasticity Buffers. Logic circuits monitor the data stream and look for the Start of Frame Delimiter (SFD). When this has been detected data is loaded into the elasticity buffer for later transmission. This will occur when sufficient preamble has been transmitted and certain internal state machine operations have been fulfilled.

Figure 5-3 shows two LERICs, A and B, daisy-chained together with LERIC A positioned at the top of the chain. A packet is received at port B1 of LERIC B and is then repeated by the other ports in the system. Figure 5-4 shows the functional timing diagram for this packet repetition represented by the signals shown in Figure 5-3. In this example only two ports in the system are shown, obviously the other ports also repeat the packet. It also indicates the operation of the LERICs' state machines in so far as can be seen by observing the Inter-LERIC bus. For reference, the repeater's state transitions are shown in terms of the states defined by the IEEE specification. The location (i.e., which port it is) of *PORT N* is also shown. The following section describes the repeater and Inter-LERIC bus transitions shown in Figure 5-4.



5.0 Functional Description (Continued)



TL/F/11240-10

\*Note 1: The activity shown on RX<sub>A1</sub> represents the transmitted signal on TX<sub>A1</sub> after being looped back by the attached transceiver.

FIGURE 5-4. Data Repetition

## 5.0 Functional Description (Continued)

The repeater is stimulated into activity by the data signal received by port B1. The LERICs in the system are alerted to forthcoming repeater operation by the falling edges on the  $\overline{ACKI}$ - $\overline{ACKO}$  daisy chain and the  $\overline{ACTN}$  bus signal. Following a defined start up delay the repeater moves to the *SEND PREAMBLE* state. The LERIC system utilizes the start up delay to perform port arbitration. When packet transmission begins the LERIC system enters the REPEAT state. The expected, for normal packet repetition, sequence of repeater states, *SEND PREAMBLE*, *SEND SFD* and *SEND DATA* is followed but is not visible upon the Inter-LEERIC bus. They are merged together into a single REPEAT state. This is also true for the *WAIT* and *IDLE* states, they appear as a combined Inter-LEERIC bus IDLE state.

Once a repeat operation has begun (i.e., the repeater leaves the *IDLE* state) it is required to transmit at least 96 bits of data or jam/preamble onto its network segments. If the duration of the received signal from *PORT N* is smaller than 96 bits, the repeater transitions to the *RECEIVE COLLISION* state (described later). This behavior is known as fragment extension.

After the packet data has been repeated, including the emptying of the LERICs' elasticity buffers, the LERIC performs the *Tw1* transmit recovery operation. This is performed during the *WAIT* state shown in the repeater state diagram.

### Receive Collisions

A receive collision is a collision which occurs on the network segment attached to *PORT N* (i.e., the collision is "received" in a similar manner as a data packet is received and then repeated to the other network segments). Not surprisingly, receive collision propagation follows a similar sequence of operations as is found with data repetition:

An arbitration process is performed to find *PORT N* and a preamble/jam pattern is transmitted by the repeater's other ports. When *PORT N* detects a collision on its segment the  $\overline{COLN}$  Inter-LEERIC bus signal is asserted. This forces all the LERICs in the system to transmit a preamble/jam pattern to their segments. This is important since they may be already transmitting data from their elasticity buffers. The repeater

moves to the *RECEIVE COLLISION* state when the LERICs begin to transmit the jam pattern. The repeater remains in this state until both the following conditions have been fulfilled:

1. At least 96 bits have been transmitted onto the network,
2. The activity has ended.

Under close examination the repeater specification reveals that the actual end of activity has its own permutations of conditions:

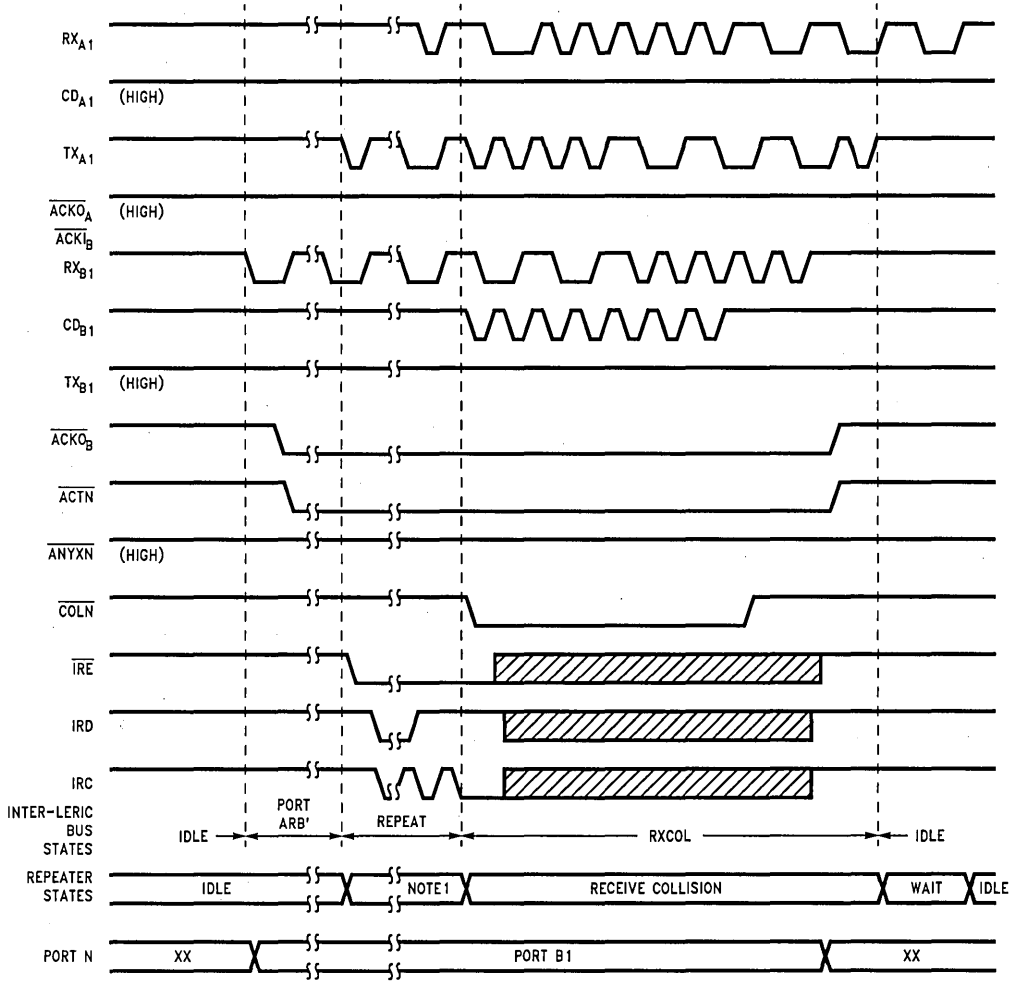
1. Collision and receive data signals may end simultaneously,
2. Receive data may appear to end before collision signals,
3. Receive data may continue for some time after the end of the collision signal.

Network segments using coaxial media may experience spurious gaps in segment activity when the collision signal goes inactive. This arises from the inter-action between the receive and collision signal squelch circuits, implemented in coaxial transceivers, and the properties of the coaxial cable itself. The repeater specification avoids propagation of these activity gaps by extending collision activity by the *Tw2* wait time. Jam pattern transmission must be sustained throughout this period. After this, the repeater will move to the *WAIT* state unless there is a data signal being received by *PORT N*.

The functional timing diagram, *Figure 5-5*, shows the operation of a repeater system during a receive collision. The system configuration is the same as earlier described and is shown in *Figure 5-3*.

The LERICs perform the same *PORT N* arbitration and data repetition operations as previously described. The system is notified of the receive collision on port B1 by the  $\overline{COLN}$  bus signal going active. This is the signal which informs the main state machines to output the jam pattern rather than the data held in the elasticity buffers. Once a collision has occurred the  $\overline{IRC}$ ,  $\overline{IRD}$  and  $\overline{IRE}$  bus signals may become undefined. When the collision has ended and the *Tw2* operation performed, the repeater moves to the *WAIT* state.

### 5.0 Functional Description (Continued)



TL/F/11240-11

\*Note 1: SEND PREAMBLE, SEND SFD, SEND DATA

FIGURE 5-5. Receive Collision

## 5.0 Functional Description (Continued)

### Transmit Collisions

A transmit collision is a collision that is detected upon a segment to which the repeater system is transmitting. The port state machine monitoring the colliding segment asserts the  $\overline{\text{ANYXN}}$  bus signal. The assertion of  $\overline{\text{ANYXN}}$  causes  $\text{PORT M}$  arbitration to begin. The repeater moves to the *TRANSMIT COLLISION* state when the port which had been  $\text{PORT N}$  starts to transmit a Manchester encoded 1 on to its network segment. While in the *TRANSMIT COLLISION* state all ports of the repeater must transmit the 1010 . . . jam pattern and  $\text{PORT M}$  arbitration is performed. Each LERIC is obliged, by the IEEE specification, to ensure all of its ports transmit for at least 96 bits once the *TRANSMIT COLLISION* state has been entered. This transmit activity is enforced by the  $\overline{\text{ANYXN}}$  bus signal. While  $\overline{\text{ANYXN}}$  is active all LERIC ports will transmit jam. To ensure this situation lasts for at least 96 bits, the MSMs inside the LERICs assert the  $\overline{\text{ANYXN}}$  signal throughout this period. After this period has elapsed,  $\overline{\text{ANYXN}}$  will only be asserted if there are multiple ports with active collisions on their network segments.

There are two possible ways for a repeater to leave the *TRANSMIT COLLISION* state. The most straightforward is when network activity (i.e., collisions and their  $T_{w2}$  extensions) end before the 96-bit enforced period expires. Under these conditions the repeater system may move directly to the *WAIT* state when 96 bits have been transmitted to all ports. If the MSM enforced period ends and there is still one port experiencing a collision the *ONE PORT LEFT* state is entered. This may be seen on the Inter-LERIC bus when  $\overline{\text{ANYXN}}$  is deasserted and  $\text{PORT M}$  stops transmitting to its network segment. In this circumstance the Inter-LERIC bus transitions to the *RECEIVE COLLISION* state. The repeater will remain in this state while  $\text{PORT M}$ 's collision,  $T_{w2}$  collision extension and any receive signals are present. When these conditions are not true, packet repetition finishes and the repeater enters the *WAIT* state.

Figure 5-6 shows a multi-LERIC system operating under transmit collision conditions. There are many different scenarios which may occur during a transmit collision, this figure illustrates one of these. The diagram begins with packet

reception by port A1. Port B1 experiences a collision, since it is not  $\text{PORT N}$  it asserts  $\overline{\text{ANYXN}}$ . This alerts the main state machines in the system to switch from data to jam pattern transmission.

Port A1 is also monitoring the  $\overline{\text{ANYXN}}$  bus line. Its assertion forces A1 to relinquish its  $\text{PORT N}$  status, start transmitting, stop asserting  $\overline{\text{ACTN}}$  and release its hold on the PSM arbitration signals ( $\overline{\text{ACKO}}$  A and  $\overline{\text{ACKI}}$  B). The first bit it transmits will be a Manchester encoded "1" in the jam pattern. Since port B1 is the only port with a collision, it attains  $\text{PORT M}$  status and stops asserting  $\overline{\text{ANYXN}}$ . It does however assert  $\overline{\text{ACTN}}$ , and exert its presence upon the PSM arbitration chain (forces  $\overline{\text{ACKO}}$  B low). The MSMs ensure that  $\overline{\text{ANYXN}}$  stays active and thus forces all of the ports, including  $\text{PORT M}$ , to transmit to their segments.

After some time port A1 experiences a collision. This arises from the presence of the packet being received from port A1's segment plus the jam signal the repeater is now transmitting onto this segment. Two packets on one segment results in a collision.  $\text{PORT M}$  now moves from B1 to A1. Port A1 fulfills the same criteria as B1 (i.e., it has an active collision on its segment), but in addition it is higher in the arbitration chain. This priority yields no benefits for port A1 since the  $\overline{\text{ANYXN}}$  signal is still active. There are now two sources driving  $\overline{\text{ANYXN}}$ , the MSMs and the collision on port B1.

Eventually the collision on port B1 ends and the  $\overline{\text{ANYXN}}$  extension by the MSMs expires. There is only one collision on the network (this may be deduced since  $\overline{\text{ANYXN}}$  is inactive) so the repeater will move to the *ONE PORT LEFT* state. The LERIC system treats this state in a similar manner to a receive collision with  $\text{PORT M}$  fulfilling the role of the receiving port. The difference from a true receive collision is that the switch from packet data to the jam pattern has already been made (controlled by  $\overline{\text{ANYXN}}$ ). Thus the state of  $\overline{\text{COLN}}$  has no effect upon repeater operations. In common with the operation of the *RECEIVE COLLISION* state, the repeater remains in this condition until the collision and receive activity on  $\text{PORT M}$  subsides. The packet repetition operation completes when the  $T_{w1}$  recovery time in the *WAIT* state has been performed.

**Note:** In transmit collision conditions  $\overline{\text{COLN}}$  will only go active if the LERIC which contained  $\text{PORT N}$  at the start of packet repetition contains  $\text{PORT M}$  during the *TRANSMIT COLLISION* and *ONE PORT LEFT* states.

5.0 Functional Description (Continued)

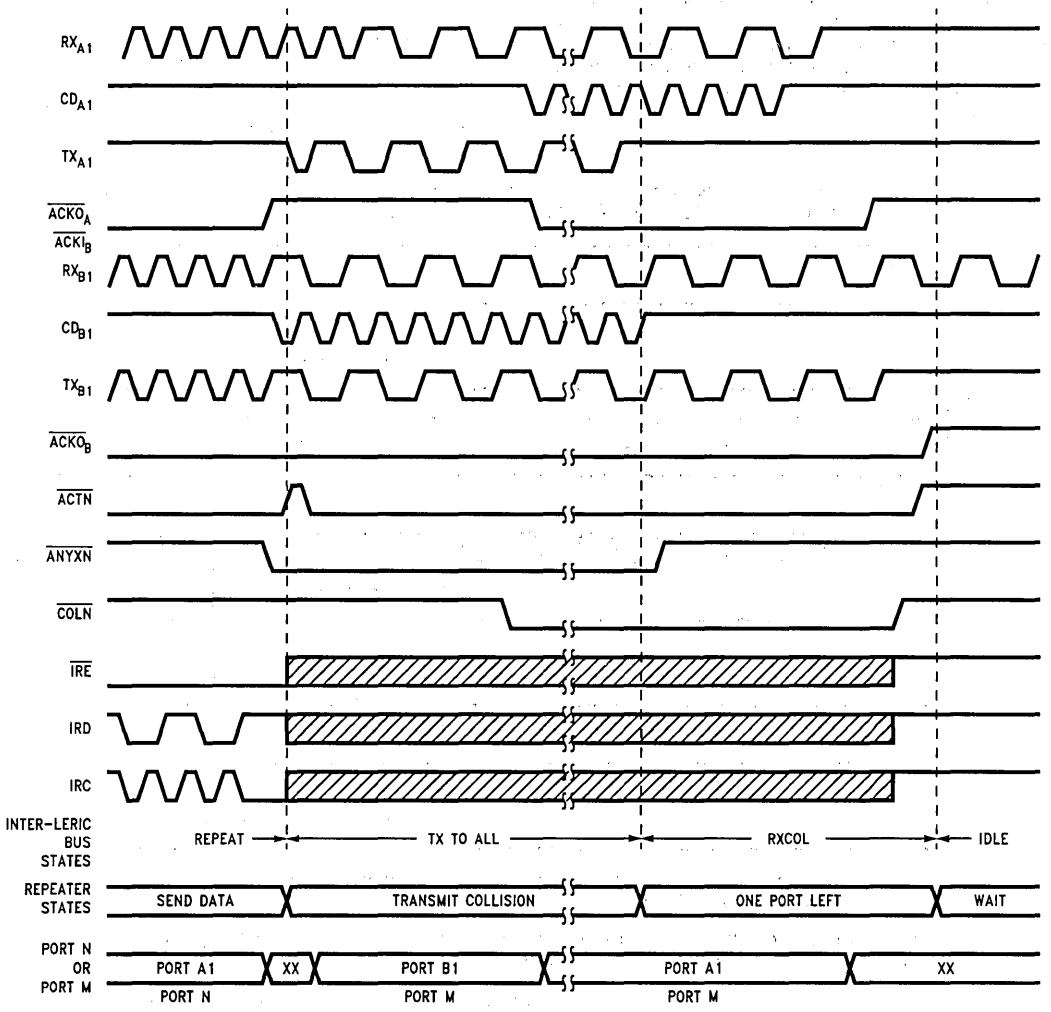


FIGURE 5-6. Transmit Collision

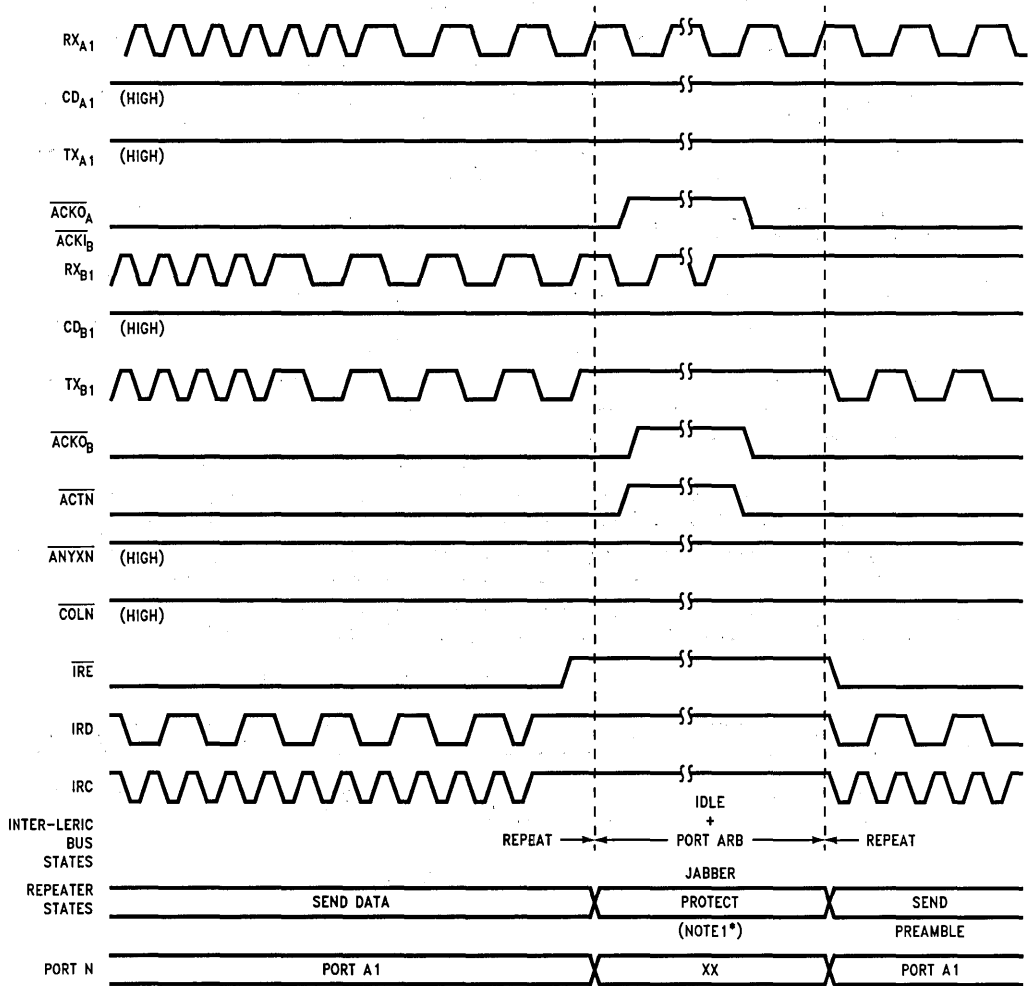
TL/F/11240-12

## 5.0 Functional Description (Continued)

### Jabber Protection

A repeater is required to disable transmit activity if the length of its current transmission reaches the jabber protect limit. This is defined by the IEEE specification's  $T_{w3}$  time. The repeater disables output for a time period defined by the  $T_{w4}$  specification, after this period normal operation may resume.

Figure 5-7 shows the effect of a jabber length packet upon a LERIC based repeater system. The **JABBER PROTECT** state is entered from the **SEND DATA** state. While the  $T_{w4}$  period is observed the Inter-LERIC bus displays the IDLE state. This is misleading since new packet activity or continuous activity (as shown in the diagram) does not result in packet repetition. This may only occur when the  $T_{w4}$  requirement has been satisfied.



TL/F/11240-13

\*Note 1: The IEEE Specification does not have a jabber protect state defined in its main state diagram, this behavior is defined in an additional MAU Jabber Lockup Protection state diagram.

FIGURE 5-7. Jabber Protect

## 5.0 Functional Description (Continued)

### 5.4 DESCRIPTION OF HARDWARE CONNECTION FOR CASCADING

#### 5.4.1 DP83955 on the Inter-LERIC Bus

When considering the hardware interface the Inter-LERIC bus may be viewed as consisting of three groups of signals:

1. Port Arbitration chain, namely:  $\overline{ACKI}$  and  $\overline{ACKO}$ . These signals are either used as point-to-point links or with external arbitration logic. In both cases the load on these signals will not be large so that the on-chip drivers are adequate.
2. Simultaneous drive and sense signals, namely:  $\overline{ACTN}$  and  $\overline{ANYXN}$ . Potentially these signals may be driven by multiple devices. It should be noticed that due to the nature of these signals, transceivers cannot be implemented for the purpose of cascading; however, bench evaluation indicates that LERICs can be cascaded together as long as the total load capacitance is 100 pF or less.
3. Drive or sense signals (i.e.,  $\overline{IRE}$ ,  $\overline{IRD}$ ,  $\overline{IRC}$  and  $\overline{COLN}$ ). Only one device asserts these signals at any instance in time. The unidirectional nature of information transfer on the  $\overline{IRE}$ ,  $\overline{IRD}$ ,  $\overline{IRC}$  and  $\overline{COLN}$  signals means a LERIC is either driving these signals or receiving them from the bus but not both at the same time. Thus a single bidirectional input/output pin is adequate for each of these signals.

#### 5.4.2 DP83956 Using the Inter-RIC Bus

When considering the hardware interface the Inter-LERIC bus may be viewed as consisting of three groups of signals:

1. Port Arbitration chain, namely:  $\overline{ACKI}$  and  $\overline{ACKO}$ . These signals are either used as point to point links or with external arbitration logic. In both cases the load on these signals will not be large so that the on-chip drivers are adequate.
2. The need for simultaneous sense and drive capabilities on the  $\overline{ACTN}$  and  $\overline{ANYXN}$  signals and the *desire to allow operation with external bus transceivers* makes it necessary for these bus signals to each have a pair of pins, one to drive the bus and the other to sense the bus. The Inter-LERIC bus on the DP83956 has been designed to connect LERICs together directly or via external bus transceivers. The latter is advantageous in large repeaters. When external bus transceivers are used they must be open collector/open drain to allow wire-ORing of the signals.
3. Drive or sense signals, i.e.,  $\overline{IRE}$ ,  $\overline{IRD}$ ,  $\overline{IRC}$  and  $\overline{COLN}$ . Only one device asserts these signals at any instance in time. The unidirectional nature of information transfer on the  $\overline{IRE}$ ,  $\overline{IRD}$ ,  $\overline{IRC}$  and  $\overline{COLN}$  signals means a LERIC is either driving these signals or receiving them from the

bus but not both at the same time. Thus a single bidirectional input/output pin is adequate for each of these signals. When an external bus transceiver is used with these signals, the Packet Enable "PKEN", an output pin of LERIC, performs the function of a drive enable and sense disable.

### 5.5 PROCESSOR AND DISPLAY INTERFACE

The processor interface pins, which include the data bus, address bus and control signals, actually perform three operations which are multiplexed on these pins. These operations are:

1. The MLOAD Operation, which performs a power up initialization cycle upon the LERIC.
2. Display Update Cycles, which are refresh operations for updating the display LEDs.
3. Processor Access Cycles, which allow  $\mu P$ 's (or simple logic) to communicate with the LERIC's registers.

These three operations are described below.

#### MLOAD Operation

The MLOAD Operation is a hardware initialization procedure performed at power on. It loads vital device configuration information into on chip configuration registers. In addition to its configuration function the  $\overline{MLOAD}$  pin is the LERIC's reset input. When  $\overline{MLOAD}$  is low all of the LERIC's repeater timers, state machines and segment partition logic are reset.

The MLOAD Operation may be accomplished by attaching the appropriate set of pull up and pull down resistors to the data and register address pins to assert logic high or low signals onto these pins, and then providing a rising edge on the  $\overline{MLOAD}$  pin as is shown in *Figure 5-8*. The mapping of chip functions to the configuration inputs is shown in Table 5-2. Such an arrangement may be performed using a simple resistor, capacitor, diode network. Performing the MLOAD Operation in this way enables the configuration of a LERIC that is in a simple repeater system (one without a processor).

Alternatively, in a complex repeater system the MLOAD Operation may be performed using a processor write cycle. This would require the  $\overline{MLOAD}$  pin be connected to the CPU's write strobe via some decoding logic, and included in the processor's memory map.



TL/F/11240-14

FIGURE 5-8. MLOAD Operation

## 5.0 Functional Description (Continued)

**TABLE 5-2. Pin Definitions for Options in the MLOAD Operation**

| Pin Name | Programming Function       | Effect When Bit Is 0 | Effect When Bit Is 1 | Function                                                                                                                                                                                                                                                                                                |                  |                                                                                                                                                                                                                                                                                                                                     |
|----------|----------------------------|----------------------|----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|          |                            |                      |                      | BYPAS2                                                                                                                                                                                                                                                                                                  | BYPAS1           | Information                                                                                                                                                                                                                                                                                                                         |
| D0<br>D1 | BYPAS1<br>BYPAS2           |                      |                      | 0<br>0<br>1<br>1                                                                                                                                                                                                                                                                                        | 0<br>1<br>0<br>1 | <p>All ports (2 to 7) use the external Transceiver Interface.</p> <p>Ports 2 and 3 use the external interface, 4 to 7 use the internal 10BASE-T transceivers.</p> <p>Ports 2 to 5 use the external interface, 6 and 7 use the internal 10BASE-T transceivers.</p> <p>All ports (2 to 7) use the internal 10BASE-T transceivers.</p> |
| D2       | Resv.                      | Not Permitted        | Required             | <p>These configuration bits select which of the repeater ports (numbers 2 to 7) are configured to use the on-chip internal 10BASE-T transceivers or the external transceiver interface. The external transceiver interface operates using AUJ compatible signal levels.</p>                             |                  |                                                                                                                                                                                                                                                                                                                                     |
| D3       | EPOLSW                     | Not Selected         | Selected             | Enables the polarity switching of the receive squelch upon detection of polarity reversal of the incoming data.                                                                                                                                                                                         |                  |                                                                                                                                                                                                                                                                                                                                     |
| D4       | Resv.                      | Not Permitted        | Required             |                                                                                                                                                                                                                                                                                                         |                  |                                                                                                                                                                                                                                                                                                                                     |
| D5       | $\overline{\text{TXONLY}}$ | Selected             | Not Selected         | This configuration bit allows the on-chip partition algorithm to restrict segment reconnection, as described in the Partition State Machine.                                                                                                                                                            |                  |                                                                                                                                                                                                                                                                                                                                     |
| D6       | $\overline{\text{CCLIM}}$  | 63                   | 31                   | The partition specification requires a port to be partitioned after a certain number of consecutive collisions. The LERIC has two values available to allow users to customize the partitioning algorithm to their environment. Please refer to the Partition State Machine, in data sheet section 7.3. |                  |                                                                                                                                                                                                                                                                                                                                     |
| D7       | MIN/MAX                    | Minimum Mode         | Maximum Mode         | The operation of the display update block is controlled by the value of this configuration bit, as described in the Display Update Cycles section.                                                                                                                                                      |                  |                                                                                                                                                                                                                                                                                                                                     |



## 5.0 Functional Description (Continued)

### 5.6 PROCESSOR AND DISPLAY INTERFACE HARDWARE CONNECTION

#### Display Update Cycles

The LERIC possesses control logic and interface pins which may be used to provide status information concerning activity on the attached network segments and the current status of repeater functions. These status cycles are completely autonomous and require only simple support circuitry to produce the data in a form suitable for a light emitting diode "LED" display. The display may be used in one of two modes:

1. Minimum Mode—General Repeater Status LEDs
2. Maximum Mode—Individual Port Status LEDs

Minimum mode, intended for simple LED displays, makes available four status indicators. The first LED denotes whether the LERIC has been forced to activate its jabber protect functions. The remaining 3 LEDs indicate if any of the LERIC's network segments are: (1) experiencing a collision, (2) receiving data, (3) currently partitioned. When minimum display mode is selected the only external components required are a 74LS374 type latch, the LEDs and their current limiting resistors.

Maximum mode differs from minimum mode by providing display information specific to individual network segments. This information denotes the collision activity, packet reception and partition status of each segment. In the case of 10BASE-T segments the link integrity status and polarity of the received data are also made available. The wide variety of information available in maximum mode may be used in its entirety or in part, thus allowing the system designer to choose the appropriate complexity of status display commensurate with the specification of the end equipment.

The signals provided and their timing relationships have been designed to interface directly with 74LS259 type addressable latches. The number of latches used being dependent upon the complexity of the display. Since the latches are octal, a pair of latches is needed to display each type of segment specific data (7 ports means 7 latch bits). The accompanying Tables 5-3 and 5-4 show the function of the interface pins in minimum and maximum modes. *Figure 5-10* shows the location of each port's status information when maximum mode is selected. This may be compared with the connection diagram (*Figure 5-9*).

Immediately following the MLOAD Operation (when the MLOAD pin transitions to a high logic state), the display logic performs an LED test operation. This operation lasts one second and while it is in effect all of the utilized LEDs will blink on. Thus an installation engineer is able to test the operation of the display by forcing the LERIC into a reset cycle (MLOAD forced low). The rising edge on the MLOAD pin starts the LED test cycle. **During the LED test cycle the LERIC does not perform packet repetition operations.**

The status display possesses a capability to lengthen the time an LED is active. At the end of the repetition of a packet, the display is frozen showing the current activity. This freezing lasts for 30 ms or until a subsequent packet is repeated. Thus at low levels of packet activity the display stretches activity information to make it discernable to the human eye. At high traffic rates the relative brightness of the LEDs indicates those segments with high or low activity.

**TABLE 5-3. Status Display Pin Functions in Minimum Mode**

| Signal Pin Name  | Mnemonic | Function in MINIMUM MODE                                                                                                                           |
|------------------|----------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| D0               | ACOL     | Provides status information indicating if there is a collision occurring on one of the segments attached to this LERIC.                            |
| D1               | AREC     | Provides status information indicating if one of this LERIC's ports is receiving a data or collision packet from a segment attached to this LERIC. |
| D2               | JAB      | Provides status information indicating that the LERIC has experienced a jabber protect condition.                                                  |
| D3               | APART    | Provides status information indicating if one of the LERIC's segments is partitioned.                                                              |
| D(7:4)           |          | No operation                                                                                                                                       |
| $\overline{STR}$ |          | This signal is the latch enable for the 374 type latch.                                                                                            |

**Note:** ACOL = Any Port Collision  
 AREC = Any Port Reception  
 JAB = Any Port Jabbering  
 APART = Port Partitioned

## 5.0 Functional Description (Continued)

**TABLE 5-4. Status Display Pin Functions in Maximum Mode**

| Signal Pin Name | Function in MAXIMUM MODE                                                                                                                                                                                                                                                                                 |
|-----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0              | Provides status information concerning the Link Integrity status of 10BASE-T segments. This signal should be connected to the data inputs of the chosen pair of 74LS259 latches.                                                                                                                         |
| D1              | Provides status information indicating if there is a collision occurring on one of the segments attached to this LERIC. This signal should be connected to the data inputs of the chosen pair of 74LS259 latches.                                                                                        |
| D2              | Provides status information indicating if one of this LERIC's ports is receiving data or a collision packet from its segment. This signal should be connected to the data inputs of the chosen pair of 74LS259 latches.                                                                                  |
| D3              | Provides status information indicating that the LERIC has experienced a jabber protect condition. Additionally, it denotes which of its ports are partitioned. This signal should be connected to the data inputs of the chosen pair of 74LS259 latches.                                                 |
| D4              | Provides status information indicating if one of this LERIC's ports is receiving data of inverse polarity. This status output is only valid if the port is configured to use its internal 10BASE-T transceiver. The signal should be connected to the data inputs of the chosen pair of 74LS259 latches. |
| D(7:5)          | These signals provide the repeater port address corresponding to the data available on D(4:0).                                                                                                                                                                                                           |
| STR             | This signal is the latch enable for the 74LS259 latches.                                                                                                                                                                                                                                                 |

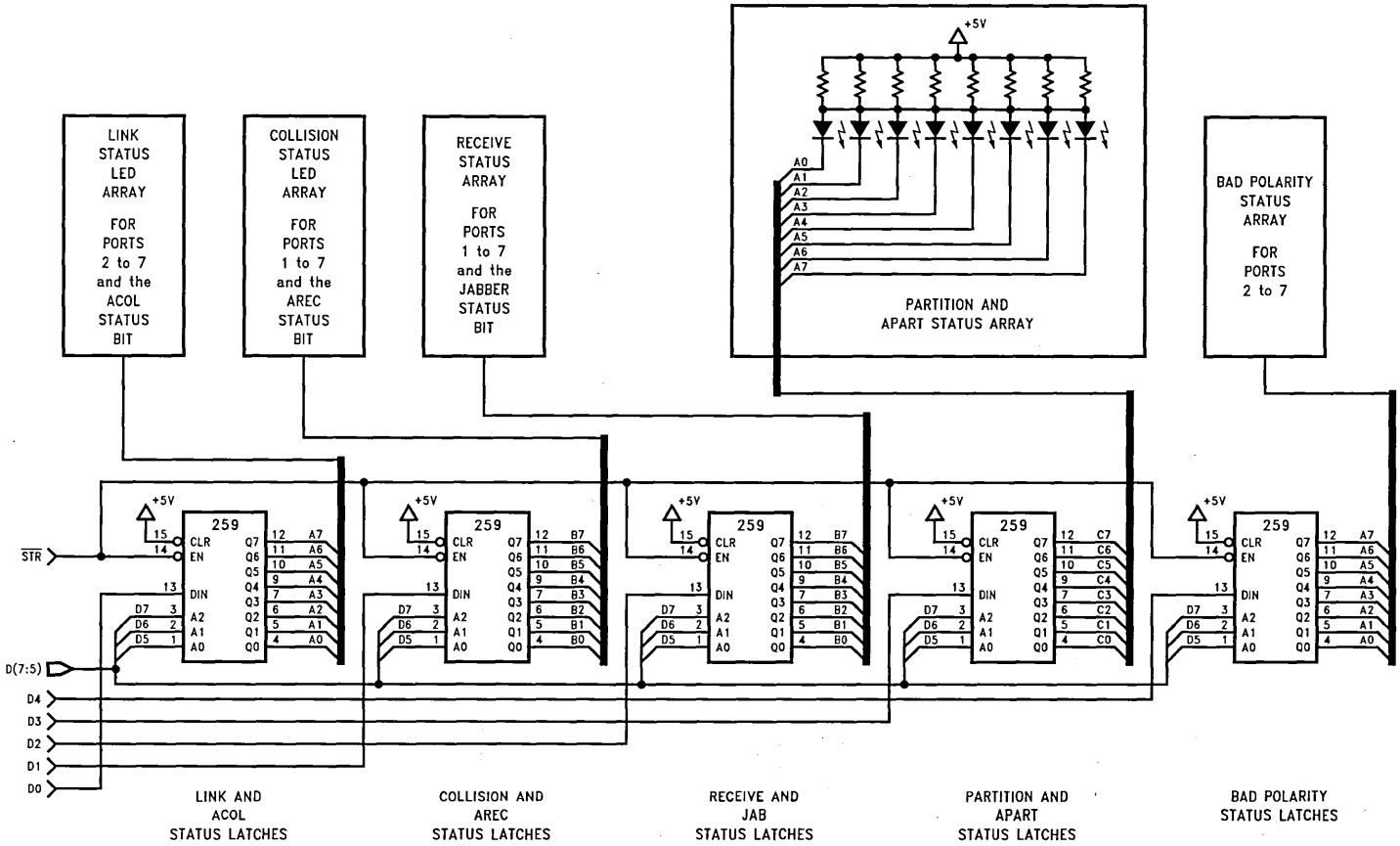


FIGURE 5-9. Maximum Mode LED Display (All Available Status Bits Used)

TL/F/11240-15

## 5.0 Functional Description (Continued)

|                   | 74LS259 Latch Inputs |         |       |       |       |       |       |       |
|-------------------|----------------------|---------|-------|-------|-------|-------|-------|-------|
| 259 Output        | Q0                   | Q1      | Q2    | Q3    | Q4    | Q5    | Q6    | A7    |
| 259 Addr S(2-0)   | 000                  | 001     | 010   | 011   | 100   | 101   | 110   | 111   |
| LERIC Port Number |                      | 1 (AU1) | 2     | 3     | 4     | 5     | 6     | 7     |
| LERIC D0          | ACOL                 |         | LINK  | LINK  | LINK  | LINK  | LINK  | LINK  |
| LERIC D1          | AREC                 | COL     | COL   | COL   | COL   | COL   | COL   | COL   |
| LERIC D2          | JAB                  | REC     | REC   | REC   | REC   | REC   | REC   | REC   |
| LERIC D3          | APART                | PART    | PART  | PART  | PART  | PART  | PART  | PART  |
| LERIC D4          |                      |         | BDPOL | BDPOL | BDPOL | BDPOL | BDPOL | BDPOL |

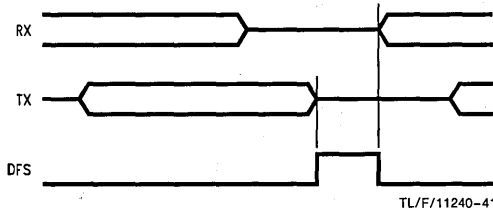
**Note:** This shows the LED Output Functions for the LED Drivers when 74LS259s are used.  
 ACOL = Any Port Collision, AREC = Any Port Reception, JAB = Any Port Jabbering,  
 LINK = Port Link, COL = Port Collision, REC = Port Reception, PART = Port Partitioned,  
 BDPOL = Bad (inverse) Polarity of received data

**FIGURE 5-10. Maximum Mode LED Definitions**

### Description of Data Freeze Strobe (DFS) Pin Operation

DFS has been implemented to assist the user to provide partial hub management statistics on a per packet per port basis. The DFS signal is asserted, active high, at the end of the transmission of each packet, and the status of that packet is frozen on the LEDs until the beginning of the next received packet or for a maximum of 30 ms as is shown in *Figure 5-11*.

The DFS signal can be used to latch the LED information into a shared buffer which acts as an external flag register, and can be used as a mechanism to trap events.



**FIGURE 5-11. DFS Operation**

### Processor Access Cycles

Access to the LERIC's on-chip registers is made via its processor interface. This utilizes a conventional non-multiplexed address (four bit) and data (four bit) bus. This bus is also used to provide data and address information to off chip LED display latches during display update cycles. While performing these cycles the LERIC behaves as a master of its data bus. Consequently a TRI-STATE bi-directional bus transceiver (e.g., 74LS245) must be placed between the LERIC and any processor bus. Internally each of the LERIC's registers is 8 bits, however there are four bits of data pins (D(3:0)). Each register is accessed on a nibble basis (4 bits at a time). D(7) of the address pins D(7:4) selects the upper and lower nibbles as described in Section 7.

To access the LERIC's registers, the processor requests a register access by asserting the read (RD) or write (WR) input strobes. The LERIC responds by finishing any current display update cycle and asserts the TRI-STATE buffer enable signal (BUFEN). If the processor cycle is a write cycle then the LERIC's buffers are disabled to prevent contention. In order to interface to the LERIC a PAL device may be used to perform the following operations:

1. Generate the LERIC's read and write strobes,
2. Control the direction signal for the 74LS245.

An example of the processor and display interfaces is shown in *Figure 5-12*.

## 5.0 Functional Description (Continued)

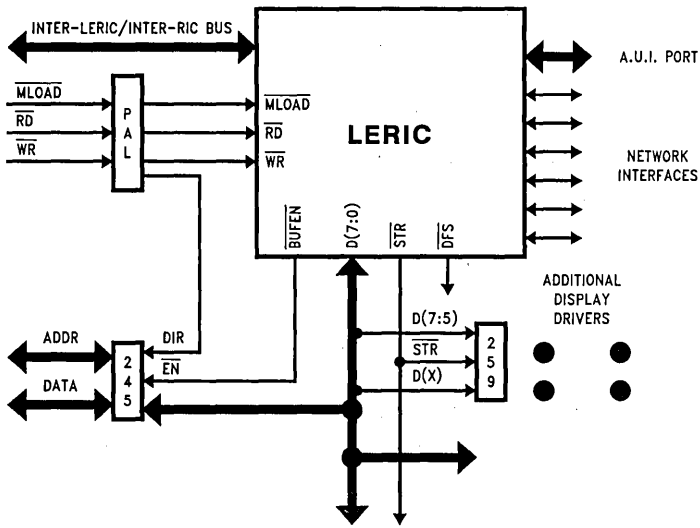


FIGURE 5-12. Processor Connection Diagram

TL/F/11240-16

## 6.0 Port Block Functions

The LERIC has 7 port logic blocks (one for each network connection). In addition to the packet repetition operations already described, the port block performs two other functions:

1. The physical connection to the network segment (transceiver function).
2. It provides a means to protect the network from malfunctioning segments (segment partition).

Each port has its own status and configuration register. This register allows the user to determine the current status of the port and configure a number of port specific functions.

### 6.1 TRANSCIEVER FUNCTIONS

The LERIC may connect to network segments in three ways:

1. Over AUI cable to transceiver boxes (Port 1)
2. Directly to board mounted transceivers.
3. To twisted pair cable via a simple interface.

The first method is only supported by LERIC Port 1 (the AUI port). Options (2) and (3) are available on Ports 2 to 7. The selection of the desired option is made at device initialization during the MLOAD operation. The Transceiver Bypass XBYPAS configuration bits are used to determine whether the ports will utilize the on-chip 10BASE-T transceivers or bypass these in favor of external transceivers. Four possible combinations of port utilization are supported (refer to Table 5-2):

1. All ports (2 to 7) use the external Transceiver Interface.
2. Ports 2 and 3 use the external interface, 4 to 7 use the internal 10BASE-T transceivers.
3. Ports 2 to 5 use the external interface, 6 and 7 use the internal 10BASE-T transceivers.
4. All ports (2 to 7) use the internal 10BASE-T transceivers.

### 10BASE-T Transceiver Operation

The LERIC contains virtually all the digital and analog circuits required for connection to 10BASE-T network segments. The only additional active component is an external driver package. The connection for a LERIC port to a 10BASE-T segment is shown in Figure 6-1. The diagram shows the components required to connect one of the LERIC's ports to a 10BASE-T segment (and lists a few module P/Ns and vendors). The major components are the driver package, a member of the 74ACT family, and an integrated filter-transformer-choke module (or discrete combination of these functions).

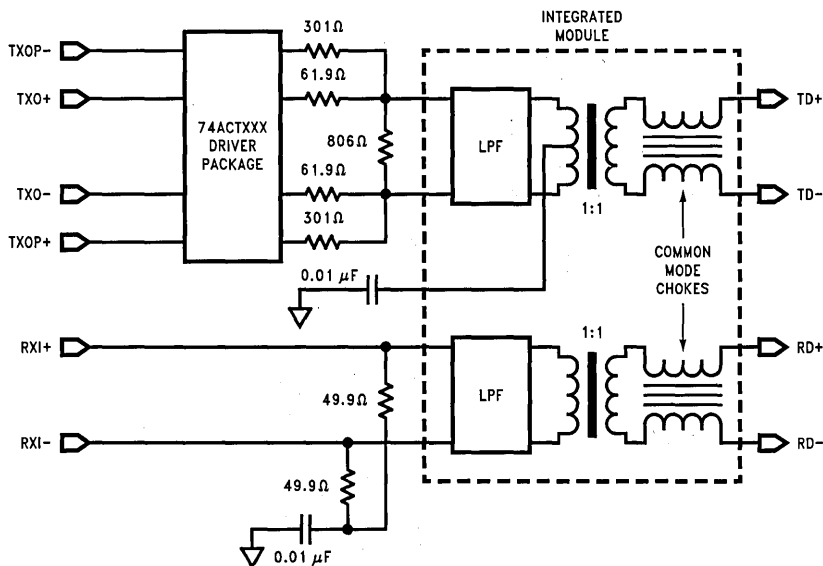
The operation of the 10BASE-T transceiver's logical functions may be modified by software control. The default mode of operation is for the transceivers to transmit and expect reception of link pulses. This may be modified if a logic one is written to the GDLNK bit of a port's status register. The port's transceiver will operate normally but will not transmit link pulses nor monitor their reception. Thus the entry to a link fail state and the associated modification of transceiver operation will not occur.

The on-chip 10BASE-T transceivers automatically detect and correct the polarity of the received data stream. This polarity detection scheme relies upon the polarity of the received link pulses and the end of packet waveform. Polarity detection and correction may be disabled through the MLOAD operation.

### External Transceiver Operation

LERIC ports 2 to 7 may be connected to media other than twisted-pair by opting to bypass the on-chip transceivers. When using external transceivers the user must perform collision detection and the other functions associated with an IEEE 802.3 Media Access Unit. Figure 6-2 shows the connection between a repeater port and a coaxial transceiver using the AUI type interface.

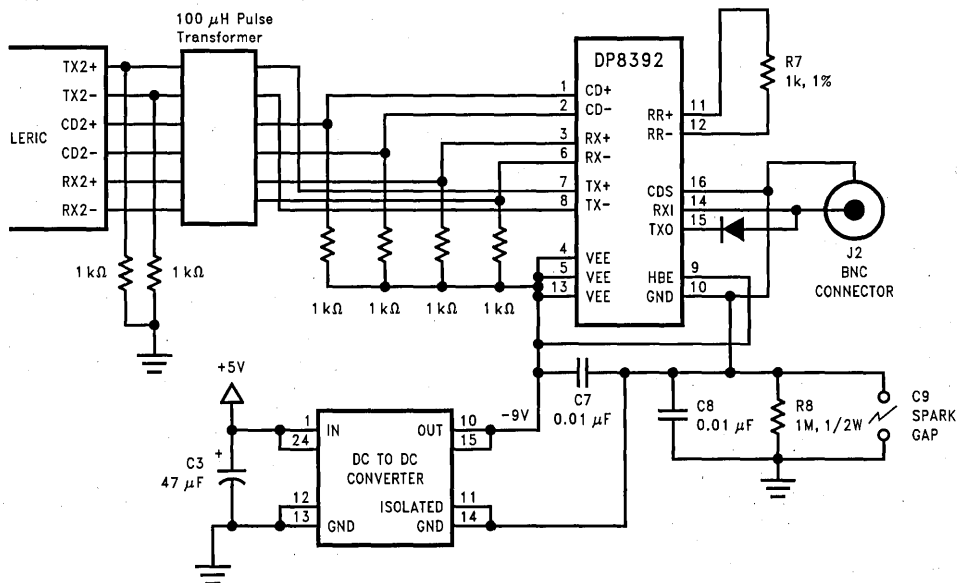
## 6.0 Port Block Functions (Continued)



TL/F/11240-17

For typical Filter-Transformer-Choke Modules refer to ETHERNET MAGNETIC VENDORS. In addition to these, the Valor FL1085 is recommended for HCT Drivers.

**FIGURE 6-1. Port Connection to a 10BASE-T Segment**



TL/F/11240-18

The above diagrams show a LERIC port (numbers 2 to 7) connected to a 10BASE-T and a 10BASE2 segment. The values of any components not indicated above are to be determined.

**FIGURE 6-2. Port Connection to a 10BASE2 Segment (AUI type Interface selected)**

## 6.0 Port Block Functions (Continued)

### 6.2 SEGMENT PARTITION

Each of the LERIC ports has a dedicated state machine to perform the functions defined by the IEEE partition algorithm as shown in *Figure 6-3*. To allow users to customize this algorithm for different applications a number of user selected options are available during device configuration at power up (the MLOAD cycle).

Two options are provided:

1. The value of consecutive counts required to partition a segment (the CCLimit specification) may be set at either 31 or 63 consecutive collisions.
2. The operation of the ports' state machines when reconnecting a segment may also be modified by the user. The Transmit Only TXONLY configuration bit allows the user to prevent segment reconnection unless the reconnecting packet is being sourced by the repeater. In this case the repeater is transmitting on to the segment rather than the segment transmitting when the repeater is idle. The normal mode of reconnection does not differentiate be-

tween such packets. The TXONLY configuration bit is input on pin D(5) during the MLOAD cycle. If this option is selected the operation of the state machine branch marked (3) in *Figure 6-3* is affected.

In addition to the autonomous operation of the partition state machines, the user may reset these state machines. This may be done individually to each port by writing a logic one to the PART bit in its status register. The port's partition state machine and associated counters are reset and the port is reconnected to the network.

### 6.3 PORT STATUS AND CONFIGURATION REGISTER FUNCTIONS

Each LERIC port has its own status and configuration register. In addition to providing status concerning the port and its network segment the register allows the following operations to be performed upon the port:

1. Port disable. When a port is disabled packet transmission and reception between the port's segment and the rest of the network is prevented.
2. Selection between normal and reduced squelch levels.

6.0 Port Block Functions (Continued)

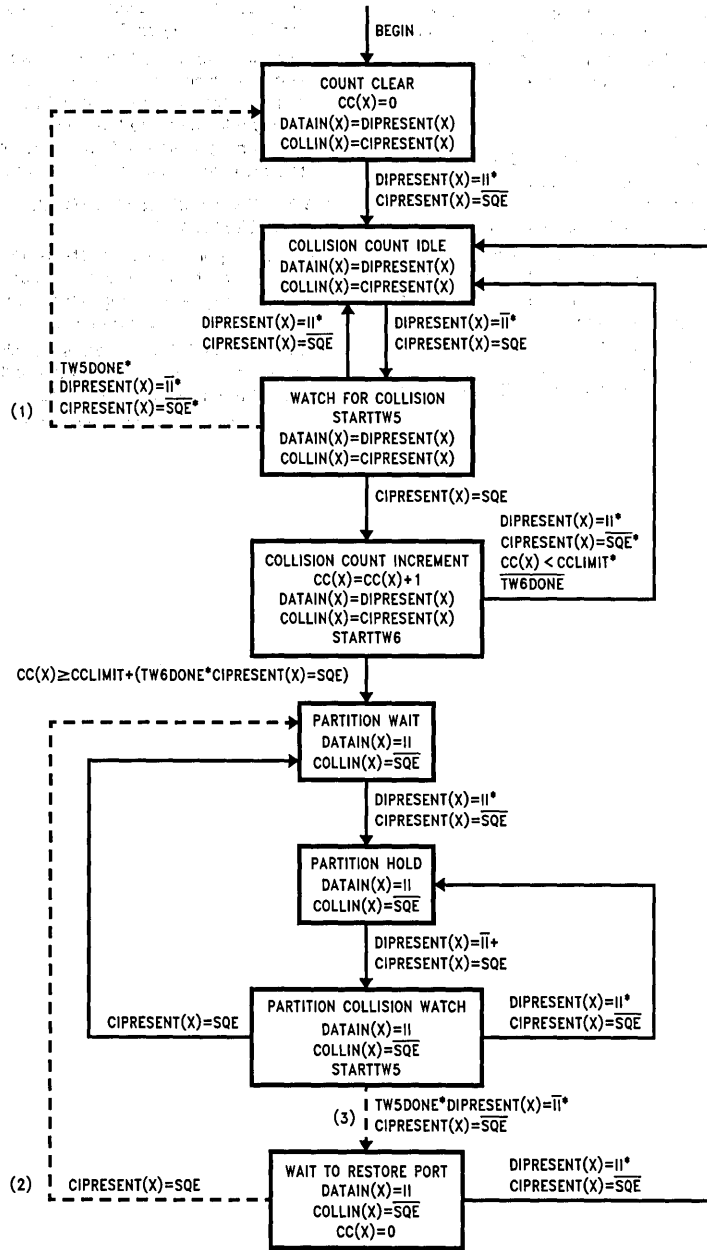


FIGURE 6-3. IEEE Segment Partition Algorithm

TL/F/11240-19



## 7.0 LERIC Registers

### 7.1 LERIC REGISTER ADDRESS MAP

The LERIC's registers may be accessed by applying the required address to the four register address (D(7:4)) input pins. Pin D(7) makes the selection between the upper and

lower nibbles of each register. The register map consists of 8 registers as shown in the Register Map in Table 7-1 which is followed by a summary of the register bits shown in Table 7-2. The definitions for these bits are shown in the detailed register definitions on the following pages.

**TABLE 7-1. Register Memory Map**

| Address D(7:4) | Name                                                                                                           |
|----------------|----------------------------------------------------------------------------------------------------------------|
| 0000<br>1000   | LERIC Status Register—Lower Nibble<br>LERIC Status Register—Upper Nibble                                       |
| 0001<br>1001   | Port 1 Status and Configuration Register—Lower Nibble<br>Port 1 Status and Configuration Register—Upper Nibble |
| 0010<br>1010   | Port 2 Status and Configuration Register—Lower Nibble<br>Port 2 Status and Configuration Register—Upper Nibble |
| 0011<br>1011   | Port 3 Status and Configuration Register—Lower Nibble<br>Port 3 Status and Configuration Register—Upper Nibble |
| 0100<br>1100   | Port 4 Status and Configuration Register—Lower Nibble<br>Port 4 Status and Configuration Register—Upper Nibble |
| 0101<br>1101   | Port 5 Status and Configuration Register—Lower Nibble<br>Port 5 Status and Configuration Register—Upper Nibble |
| 0110<br>1110   | Port 6 Status and Configuration Register—Lower Nibble<br>Port 6 Status and Configuration Register—Upper Nibble |
| 0111<br>1111   | Port 7 Status and Configuration Register—Lower Nibble<br>Port 7 Status and Configuration Register—Upper Nibble |

**Register Array Bit Map**

| Address D(7:4) | D(3)                              | D(2)                            | D(1)                             | D(0)                              |
|----------------|-----------------------------------|---------------------------------|----------------------------------|-----------------------------------|
| 0000<br>1000   | $\overline{\text{PART}}$<br>Resv  | JAB<br>Resv                     | $\overline{\text{AREC}}$<br>Resv | $\overline{\text{ACOL}}$<br>Resv  |
| 0001<br>1001   | $\overline{\text{PART}}$<br>DISPT | $\overline{\text{REC}}$<br>Resv | $\overline{\text{COL}}$<br>Resv  | $\overline{\text{GDLNK}}$<br>Resv |
| 0010<br>1010   | $\overline{\text{PART}}$<br>DISPT | $\overline{\text{REC}}$<br>Resv | $\overline{\text{COL}}$<br>POL   | $\overline{\text{GDLNK}}$<br>SQL  |
| 0011<br>1011   | $\overline{\text{PART}}$<br>DISPT | $\overline{\text{REC}}$<br>Resv | $\overline{\text{COL}}$<br>POL   | $\overline{\text{GDLNK}}$<br>SQL  |
| 0100<br>1100   | $\overline{\text{PART}}$<br>DISPT | $\overline{\text{REC}}$<br>Resv | $\overline{\text{COL}}$<br>POL   | $\overline{\text{GDLNK}}$<br>SQL  |
| 0101<br>1101   | $\overline{\text{PART}}$<br>DISPT | $\overline{\text{REC}}$<br>Resv | $\overline{\text{COL}}$<br>POL   | $\overline{\text{GDLNK}}$<br>SQL  |
| 0110<br>1110   | $\overline{\text{PART}}$<br>DISPT | $\overline{\text{REC}}$<br>Resv | $\overline{\text{COL}}$<br>POL   | $\overline{\text{GDLNK}}$<br>SQL  |
| 0111<br>1111   | $\overline{\text{PART}}$<br>DISPT | $\overline{\text{REC}}$<br>Resv | $\overline{\text{COL}}$<br>POL   | $\overline{\text{GDLNK}}$<br>SQL  |

## 7.0 LERIC Registers (Continued)

### 7.2 LERIC STATUS REGISTER

This register contains real time information concerning the operation of the LERIC.

| D(3) | D(2) | D(1) | D(0) | D(3)  | D(2) | D(1) | D(0) |
|------|------|------|------|-------|------|------|------|
| Resv | Resv | Resv | Resv | APART | JAB  | AREC | ACOL |

| Symbol | Bit  | R/W | Description                                                                                                                                                                                  |
|--------|------|-----|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ACOL   | D(0) | R   | Any Collisions<br>0: A collision is occurring at one or more of the LERIC's ports<br>1: No collisions                                                                                        |
| AREC   | D(1) | R   | Any Receive<br>0: One of the LERIC's ports is the current packet or collision receiver<br>1: No packet or collision reception within this LERIC                                              |
| JAB    | D(2) | R   | Jabber Protect<br>0: The LERIC has been forced into jabber protect state by one of its ports or by another port on the Inter-LERIC bus (operations)<br>1: No jabber protect conditions exist |
| APART  | D(3) | R   | Any Partition<br>0: One or more ports are partitioned<br>1: No ports are partitioned                                                                                                         |
| Resv   | D(0) | R   | Reserved for future use<br>Value set at logic one                                                                                                                                            |
| Resv   | D(1) | R   | Reserved for future use<br>Value set at logic one                                                                                                                                            |
| Resv   | D(2) | R   | Reserved for future use<br>Value set at logic one                                                                                                                                            |
| Resv   | D(3) | R   | Reserved for future use<br>Value set at logic one                                                                                                                                            |

## 7.0 LERIC Registers (Continued)

### 7.3 PORT STATUS AND CONFIGURATION REGISTERS

|       |      |      |      |      |      |      |       |
|-------|------|------|------|------|------|------|-------|
| D(3)  | D(2) | D(1) | D(0) | D(3) | D(2) | D(1) | D(0)  |
| DISPT | Resv | POL  | SQL  | PART | REC  | COL  | GDLNK |

| Symbol | Bit  | R/W | Description                                                                                                                                                                                                                                                                                                                                                                                       |
|--------|------|-----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| GDLNK  | D(0) | R/W | <p>Good Link</p> <p>0: Link pulses are being received by the port</p> <p>1: Link pulses are not being received by the port logic</p> <p><b>Note:</b> Writing a 1 to this bit will cause the 10BASE-T transceiver not to transmit or monitor the reception of link pulses. If the internal 10BASE-T transceivers are not selected or if port 1 (AUI port) is read, then this bit is undefined.</p> |
| COL    | D(1) | R   | <p>Collision</p> <p>0: A collision is happening or has occurred during the current packet</p> <p>1: No collisions have occurred as yet during this packet</p>                                                                                                                                                                                                                                     |
| REC    | D(2) | R   | <p>Receive</p> <p>0: This port is now or has been the receive source of packet or collision information for the current packet.</p> <p>1: This port has not been the receive source during the current packet</p>                                                                                                                                                                                 |
| PART   | D(3) | R/W | <p>Partition</p> <p>0: This port is partitioned</p> <p>1: This port is not partitioned</p> <p>Writing a logic one to this bit forces segment reconnection and partition state machine reset. Writing a zero to this bit has no effect.</p>                                                                                                                                                        |
| SQL    | D(0) | R/W | <p>Squelch Level</p> <p>0: Port operates with normal IEEE receive squelch level</p> <p>1: Port operates with reduced receive squelch levels</p> <p><b>Note:</b> This bit has no effect when the external transceiver is selected.</p>                                                                                                                                                             |
| POL    | D(1) | R   | <p>Polarity</p> <p>0: Polarity is not inverted</p> <p>1: Polarity is inverted</p>                                                                                                                                                                                                                                                                                                                 |
| Resv   | D(2) | R   | <p>"Reserved"</p> <p>"Value set to logic zero"</p>                                                                                                                                                                                                                                                                                                                                                |
| DISPT  | D(3) | R/W | <p>Disable Port</p> <p>0: Port operates as defined by repeater operations</p> <p>1: All port activity is prevented</p>                                                                                                                                                                                                                                                                            |

## 8.0 Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

|                                 |                          |
|---------------------------------|--------------------------|
| Supply Voltage ( $V_{CC}$ )     | 0.5V to 7.0V             |
| DC Input Voltage ( $V_{IN}$ )   | -0.5V to $V_{CC} + 0.5V$ |
| DC Output Voltage ( $V_{OUT}$ ) | -0.5V to $V_{CC} + 0.5V$ |

|                                                       |                 |
|-------------------------------------------------------|-----------------|
| Storage Temperature Range ( $T_{STG}$ )               | -65°C to +150°C |
| Power Dissipation ( $P_D$ )                           | 1.5W            |
| Lead Temperature ( $T_L$ )<br>(Soldering, 10 Seconds) | 260°C           |
| ESD Rating<br>( $R_{ZAP} = 1.5k, C_{ZAP} = 120 pF$ )  | 1.5 kV          |

## 9.0 DC Specifications $T_A = 0^\circ\text{C}$ to $+70^\circ\text{C}$ , $V_{CC} = 5V \pm 5\%$ unless otherwise specified

| Symbol                                                               | Description                                                          | Conditions                                         | Min       | Max              | Units         |
|----------------------------------------------------------------------|----------------------------------------------------------------------|----------------------------------------------------|-----------|------------------|---------------|
| <b>PROCESSOR, LED, TWISTED-PAIR PORTS AND INTER-LERIC INTERFACES</b> |                                                                      |                                                    |           |                  |               |
| $V_{OH}$                                                             | Minimum High Level Output Voltage                                    | $I_{OH} = -8 \text{ mA}$                           | 3.5       |                  | V             |
| $V_{OL}$                                                             | Minimum Low Level Output Voltage                                     | $I_{OL} = 8 \text{ mA}$                            |           | 0.4              | V             |
| $V_{IH}$                                                             | Minimum High Level Input Voltage                                     |                                                    | 2.0       |                  | V             |
| $V_{IL}$                                                             | Maximum Low Level Input Voltage                                      |                                                    |           | 0.8              | V             |
| $I_{IN}$                                                             | Input Current                                                        | $V_{IN} = V_{CC}$ or GND                           | -1.0      | 1.0              | $\mu\text{A}$ |
| $I_{OZ}$                                                             | Maximum TRI-STATE Output Leakage Current                             | $V_{OUT} = V_{CC}$ or GND                          | -10       | 10               | $\mu\text{A}$ |
| $I_{CC}$                                                             | Average Supply Current                                               | $V_{IN} = V_{CC}$ or GND<br>$V_{CC} = 5.25V$       |           | 250              | mA            |
| <b>AUI (PORT 1)</b>                                                  |                                                                      |                                                    |           |                  |               |
| $V_{OD}$                                                             | Differential Output Voltage ( $TX \pm$ )                             | 78 $\Omega$ Termination and 270 $\Omega$ Pulldowns | $\pm 550$ | $\pm 1200$       | mV            |
| $V_{OB}$                                                             | Differential Output Voltage Imbalance ( $TX \pm$ )                   | 78 $\Omega$ Termination and 270 $\Omega$ Pulldowns |           | 40 mV<br>Typical |               |
| $V_U$                                                                | Undershoot Voltage ( $TX \pm$ )                                      | 78 $\Omega$ Termination and 270 $\Omega$ Pulldowns |           | 80 mV<br>Typical |               |
| $V_{DS}$                                                             | Differential Squelch Threshold ( $RX \pm, CD \pm$ )                  |                                                    | -175      | -300             | mV            |
| $V_{CM}$                                                             | Differential Input Common Mode Voltage ( $RX \pm, CD \pm$ ) (Note 1) |                                                    | 0         | 5.5              | V             |

## 9.0 DC Specifications $T_A = 0^\circ\text{C}$ to $+70^\circ\text{C}$ , $V_{CC} = 5\text{V} \pm 5\%$ unless otherwise specified (Continued)

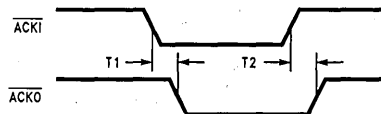
| Symbol                        | Description                                                            | Conditions                                          | Min       | Max           | Units |
|-------------------------------|------------------------------------------------------------------------|-----------------------------------------------------|-----------|---------------|-------|
| <b>PSEUDO AUI (PORTS 2-7)</b> |                                                                        |                                                     |           |               |       |
| $V_{POD}$                     | Differential Output Voltage (TX $\pm$ )                                | 270 $\Omega$ Termination and 1 k $\Omega$ Pulldowns | $\pm 450$ | $\pm 1200$    | mV    |
| $V_{POB}$                     | Differential Output Voltage Imbalance (TX $\pm$ )                      | 270 $\Omega$ Termination and 1 k $\Omega$ Pulldowns |           | 40 mV Typical |       |
| $V_{PU}$                      | Undershoot Voltage (TX $\pm$ )                                         | 270 $\Omega$ Termination and 1 k $\Omega$ Pulldowns |           | 80 mV Typical |       |
| $V_{PDS}$                     | Differential Squelch Threshold (RX $\pm$ , CD $\pm$ )                  |                                                     | -175      | -300          | mV    |
| $V_{PCM}$                     | Differential Input Common Mode Voltage (RX $\pm$ , CD $\pm$ ) (Note 1) |                                                     | 0         | 5.5           | V     |

|                                 |                                                                   |  |                        |                        |          |
|---------------------------------|-------------------------------------------------------------------|--|------------------------|------------------------|----------|
| <b>TWISTED-PAIR (PORTS 2-7)</b> |                                                                   |  |                        |                        |          |
| $V_{RON}$                       | Minimum Receive Squelch Threshold:<br>Normal Mode<br>Reduced Mode |  | $\pm 300$<br>$\pm 175$ | $\pm 585$<br>$\pm 300$ | mV<br>mV |

**Note 1:** This parameter is guaranteed by design and is not tested.

## 10.0 Switching Characteristics

### PORT ARBITRATION TIMING



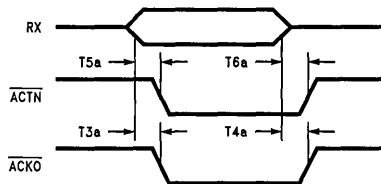
TL/F/11240-20

| Symbol     | Number | Parameter                                                      | Min | Max | Units |
|------------|--------|----------------------------------------------------------------|-----|-----|-------|
| ackilackol | T1     | $\overline{\text{ACKI}}$ Low to $\overline{\text{ACKO}}$ Low   |     | 26  | ns    |
| ackihackoh | T2     | $\overline{\text{ACKI}}$ High to $\overline{\text{ACKO}}$ High |     | 23  | ns    |

**Note:** Timing valid with no receive or collision activities.

### RECEIVE TIMING—AUI PORTS

Receive activity propagation start up and end delays for ports in non 10BASE-T mode



TL/F/11240-21

| Symbol  | Number | Parameter                                             | Min | Max | Units |
|---------|--------|-------------------------------------------------------|-----|-----|-------|
| rxackol | T3a    | RX Active to $\overline{\text{ACKO}}$ Low             |     | 66  | ns    |
| rxackoh | T4a    | RX Inactive to $\overline{\text{ACKO}}$ High (Note 1) |     | 235 | ns    |
| rxactnl | T5a    | RX Active to $\overline{\text{ACTN}}$ Low             |     | 75  | ns    |
| rxactnh | T6a    | RX Inactive to $\overline{\text{ACTN}}$ High (Note 1) |     | 235 | ns    |

**Note:**  $\overline{\text{ACKI}}$  assumed high

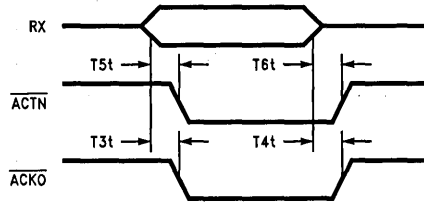
**Note 1:** This time includes EOP.

**Note 2:** This parameter assumes squelch triggers on negative edge of RX data.

## 10.0 Switching Characteristics (Continued)

### RECEIVE TIMING-10BASE-T PORTS

Receive activity propagation start up and end delays for ports in 10BASE-T mode



TL/F/11240-22

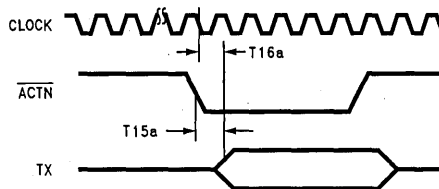
| Symbol  | Number | Parameter                                             | Min | Max | Units |
|---------|--------|-------------------------------------------------------|-----|-----|-------|
| rxackol | T3t    | RX Active to $\overline{\text{ACKO}}$ Low             |     | 300 | ns    |
| rxackoh | T4t    | RX Inactive to $\overline{\text{ACKO}}$ High (Note 1) |     | 280 | ns    |
| rxactnl | T5t    | RX Active to $\overline{\text{ACTN}}$ Low             |     | 300 | ns    |
| rxactnh | T6t    | RX Inactive to $\overline{\text{ACTN}}$ High (Note 1) |     | 280 | ns    |

Note:  $\overline{\text{ACKI}}$  assumed high.

Note 1: This time includes EOP.

### TRANSMIT TIMING—AUI PORTS

Transmit activity propagation start up and end delays for ports in non 10BASE-T mode



TL/F/11240-23

| Symbol   | Number | Parameter                                 | Min | Max | Units |
|----------|--------|-------------------------------------------|-----|-----|-------|
| actnltxa | T15a   | $\overline{\text{ACTN}}$ Low to TX Active |     | 675 | ns    |
| clkitxa  | T16a   | CLOCK in to TX Active (Note 1)            |     | 45  | ns    |

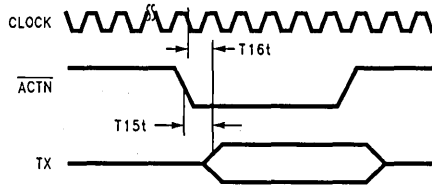
Note:  $\overline{\text{ACKI}}$  assumed high.

Note 1: Measurement from previous falling edge of the clock.

## 10.0 Switching Characteristics (Continued)

### TRANSMIT TIMING—10BASE-T PORTS

Receive activity propagation start up and end delays for ports in 10BASE-T mode



TL/F/11240-24

| Symbol   | Number | Parameter                      | Min | Max | Units |
|----------|--------|--------------------------------|-----|-----|-------|
| actnltxa | T15t   | ACTN Low to TX Active          |     | 790 | ns    |
| clkitxa  | T16t   | CLOCK in to TX Active (Note 1) |     | 45  | ns    |

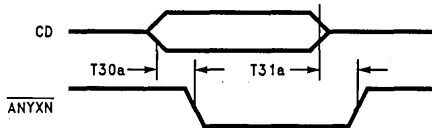
Note:  $\overline{ACK}$  assumed high.

Note 1: Clock not drawn to scale. In this measurement, falling edge of the clock for even ports and rising edge of the clock for odd ports are considered.

### COLLISION TIMING—AUI PORTS

Collision activity propagation start up and end delays for ports in non 10BASE-T mode

#### TRANSMIT COLLISION TIMING



TL/F/11240-25

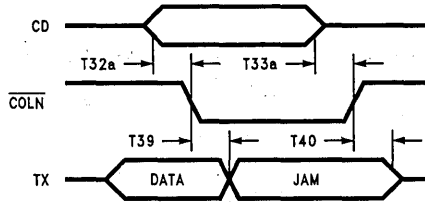
| Symbol         | Number | Parameter                              | Min | Max | Units |
|----------------|--------|----------------------------------------|-----|-----|-------|
| cdaanyxnl      | T30a   | CD Active to ANYXN Low                 |     | 85  | ns    |
| cdianyxn timer | T31a   | CD Inactive to ANYXN High (Notes 1, 2) |     | 285 | ns    |

Note 1: TX collision extension has already been performed and no other port is driving ANYXN.

Note 2: Includes TW2.

## 10.0 Switching Characteristics (Continued)

### RECEIVE COLLISION TIMING



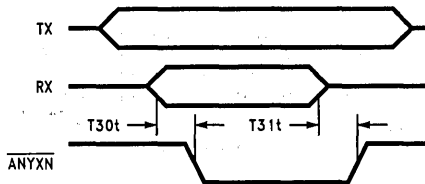
TL/F/11240-26

| Symbol   | Number | Parameter                                           | Min | Max | Units |
|----------|--------|-----------------------------------------------------|-----|-----|-------|
| cdacolna | T32a   | CD Active to $\overline{\text{COLN}}$ Low           |     | 75  | ns    |
| cdicolni | T33a   | CD Inactive to $\overline{\text{COLN}}$ High        |     | 215 | ns    |
| colnls   | T39    | $\overline{\text{COLN}}$ Low to Start of JAM        |     | 400 | ns    |
| colnhje  | T40    | $\overline{\text{COLN}}$ High to End of JAM(Note 1) |     | 585 | ns    |

Note 1: Reception ended before  $\overline{\text{COLN}}$  goes high.

### COLLISION TIMING—10BASE-T PORTS

Collision activity propagation start up and end delays for ports in 10BASE-T mode



TL/F/11240-27

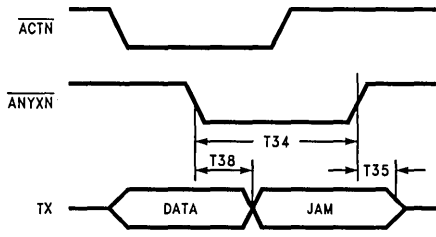
| Symbol   | Number | Parameter                                                     | Min | Max | Units |
|----------|--------|---------------------------------------------------------------|-----|-----|-------|
| colaanyl | T30t   | Collision Active to $\overline{\text{ANYXN}}$ Low             |     | 800 | ns    |
| colianyh | T31t   | Collision Inactive to $\overline{\text{ANYXN}}$ High (Note 1) |     | 450 | ns    |

Note 1: TX collision extension has already been performed and no other port is asserting  $\overline{\text{ANYXN}}$ .



# 10.0 Switching Characteristics (Continued)

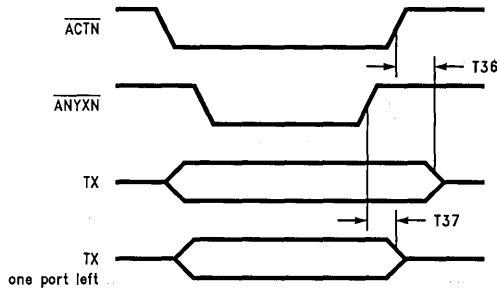
## COLLISION TIMING—ALL PORTS



TL/F/11240-28

| Symbol   | Number | Parameter                        | Min | Max | Units |
|----------|--------|----------------------------------|-----|-----|-------|
| anylmin  | T34    | ANYXN Low Time                   | 96  |     | bits  |
| anyhtxai | T35    | ANYXN High to TX to All Inactive | 20  | 370 | ns    |
| anylsj   | T38    | ANYXN Low to Start of JAM        |     | 565 | ns    |

## COLLISION TIMING—ALL PORTS



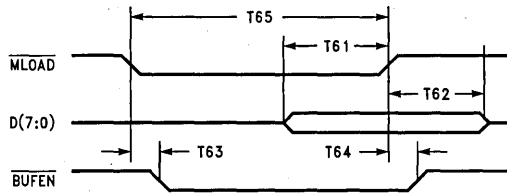
TL/F/11240-29

| Symbol   | Number | Parameter                                 | Min | Max | Units |
|----------|--------|-------------------------------------------|-----|-----|-------|
| actnhtxi | T36    | ACTN High to TX Inactive                  |     | 410 | ns    |
| anyhtxoi | T37    | ANYXN High to TX "One Port Left" Inactive | 20  | 200 | ns    |

Note: 96 bits of JAM have already been propagated.

## 10.0 Switching Characteristics (Continued)

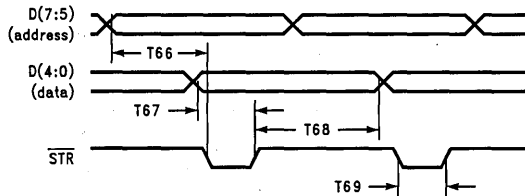
### RESET TIMING



TL/F/11240-30

| Symbol   | Number | Parameter                                                        | Min | Max | Units |
|----------|--------|------------------------------------------------------------------|-----|-----|-------|
| resdats  | T61    | Data Setup                                                       | 20  |     | ns    |
| resdath  | T62    | Data Hold                                                        | 20  |     | ns    |
| reslbufh | T63    | $\overline{\text{MLOAD}}$ Low to $\overline{\text{BUFEN}}$ Low   |     | 35  | ns    |
| reshbufh | T64    | $\overline{\text{MLOAD}}$ High to $\overline{\text{BUFEN}}$ High |     | 35  | ns    |
| resw     | T65    | $\overline{\text{MLOAD}}$ Width                                  | 800 |     | ns    |

### LED STROBE TIMING

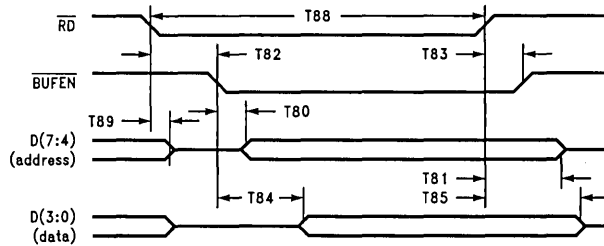


TL/F/11240-31

| Symbol  | Number | Parameter            | Min | Max | Units |
|---------|--------|----------------------|-----|-----|-------|
| strads  | T66    | Strobe Address Setup | 70  | 100 | ns    |
| strdats | T67    | Strobe Data Setup    | 35  | 55  | ns    |
| strdath | T68    | Strobe Data Hold     | 145 | 165 | ns    |
| strw    | T69    | Strobe Width         | 30  | 65  | ns    |

# 10.0 Switching Characteristics (Continued)

## REGISTER READ TIMING

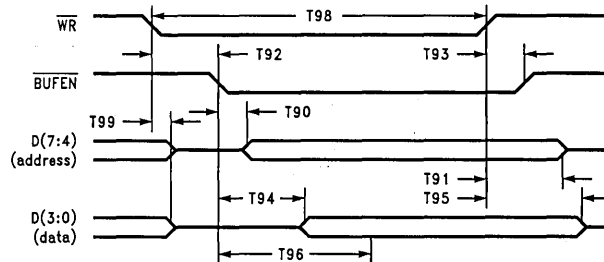


TL/F/11240-32

| Symbol   | Number | Parameter                                       | Min | Max | Units |
|----------|--------|-------------------------------------------------|-----|-----|-------|
| rdadrs   | T80    | Address Setup from $\overline{BUFEN}$ Low       | 0   | 85  | ns    |
| rdadrh   | T81    | Address Hold after $\overline{RD}$ High         | 0   |     | ns    |
| rdlbufi  | T82    | $\overline{RD}$ Low to $\overline{BUFEN}$ Low   | 80  | 355 | ns    |
| rdhbufh  | T83    | $\overline{RD}$ High to $\overline{BUFEN}$ High |     | 35  | ns    |
| bufldatv | T84    | $\overline{BUFEN}$ Low to Data Valid            |     | 190 | ns    |
| rddath   | T85    | Read Data Hold                                  | 60  |     | ns    |
| rdw      | T88    | $\overline{RD}$ Width                           | 650 |     | ns    |
| rdtr     | T89    | $\overline{RD}$ Low to $D(7:4)$ TRI-STATE       | 80  | 355 | ns    |

Note: Minimum high time between read/write cycles is 100 ns.

## REGISTER WRITE TIMING



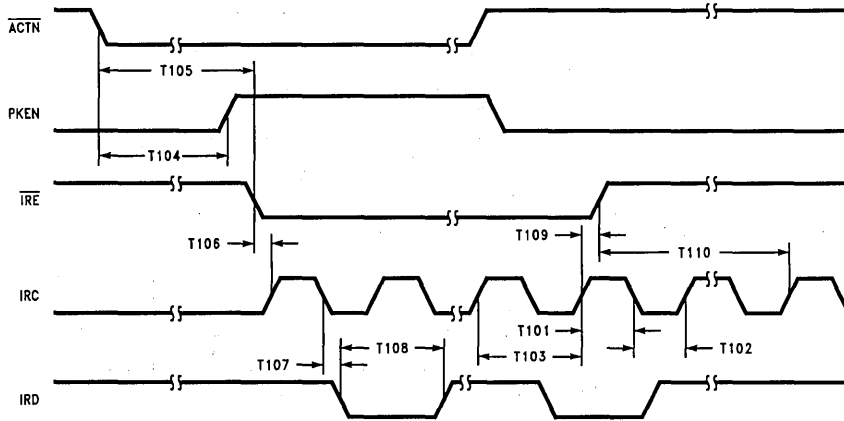
TL/F/11240-33

| Symbol  | Number | Parameter                                       | Min | Max | Units |
|---------|--------|-------------------------------------------------|-----|-----|-------|
| wradrs  | T90    | Address Setup from $\overline{BUFEN}$ Low       | 0   | 14  | ns    |
| wradrh  | T91    | Address Hold after $\overline{WR}$ High         | 0   |     | ns    |
| wrlbufi | T92    | $\overline{WR}$ Low to $\overline{BUFEN}$ Low   | 80  | 355 | ns    |
| wrhbufh | T93    | $\overline{WR}$ High to $\overline{BUFEN}$ High |     | 35  | ns    |
| wradatv | T94    | $\overline{BUFEN}$ Low to Data Valid            |     | 160 | ns    |
| wrdath  | T95    | Write Data Hold                                 | 0   |     | ns    |
| wrdatr  | T96    | $\overline{BUFEN}$ Low to Data Latched          | 245 |     | ns    |
| wrw     | T98    | $\overline{WR}$ Width                           | 650 |     | ns    |
| wrtr    | T99    | $\overline{WR}$ Low to $D(7:0)$ TRI-STATE       | 80  | 355 | ns    |

Note: Minimum high time between read/write cycles is 100 ns.

## 10.0 Switching Characteristics (Continued)

### INTER-LERIC BUS OUTPUT TIMING

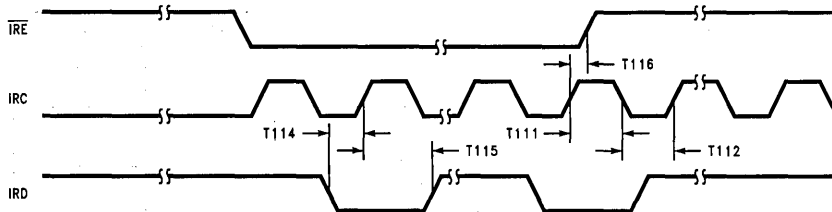


TL/F/11240-34

| Symbol      | Number | Parameter                                  | Min | Max | Units  |
|-------------|--------|--------------------------------------------|-----|-----|--------|
| ircoh       | T101   | IRC Output High Time                       | 40  | 60  | %      |
| ircol       | T102   | IRC Output Low Time                        | 40  | 60  | %      |
| ircoc       | T103   | IRC Output Cycle Time                      | 90  | 110 | ns     |
| actndapkena | T104   | ACTNd Active to PKEN Active (Note 1)       | 500 |     | ns     |
| actnolireol | T105   | ACTN Output Low to IRE Output Low          | 500 |     | ns     |
| ireolirca   | T106   | IRE Output Low to First Rising Edge of IRC |     | 1.8 | μs     |
| irdov       | T107   | IRD Output Valid from IRC                  |     | 10  | ns     |
| irdos       | T108   | IRD Output Stable Valid Time               | 90  |     | ns     |
| ircohireh   | T109   | IIRC Output High to IRE High               | 30  | 70  | ns     |
| ircclks     | T110   | Number of IRCs after IRE High              | 5   | 5   | clocks |

Note 1: This parameter applies to DP83956 only.

### INTER-LERIC BUS INPUT TIMING



TL/F/11240-35

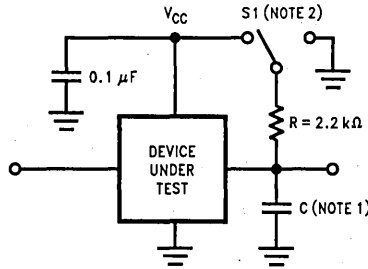
| Symbol    | Number | Parameter                  | Min | Max | Units |
|-----------|--------|----------------------------|-----|-----|-------|
| ircih     | T111   | IRC Input High Time        | 20  |     | ns    |
| ircil     | T112   | IRC Input Low Time         | 20  |     | ns    |
| irdisirc  | T114   | IRD Input Setup to IRC     | 5   |     | ns    |
| irdihirc  | T115   | IRD Input Hold from IRC    | 10  |     | ns    |
| ircihireh | T116   | IRC Input High to IRE High | 25  | 75  | ns    |

### 11.0 AC Timing Test Conditions

All specifications are valid only if the mandatory isolation is employed and all differential signals are taken to be at AUI side of the pulse transformer.

Input Pulse Levels (TTL/CMOS) GND to 3.0V  
 Input Rise and Fall Times (TTL/CMOS) 5 ns  
 Input and Output Reference Levels (TTL/CMOS) 1.5V

Input Pulse Levels (Diff.) -350 mV to -1315 mV  
 Input and Output Reference Levels (Diff.) 50% Point of the Differential  
 TRI-STATE Reference Levels Float ( $\Delta V$ )  $\pm 0.5V$   
 Output Load (See Figure Below)



TL/F/11240-42

**Note 1:** 100 pF, include scope and jig capacitance.

**Note 2:** S1 = Open for timing tests for push pull outputs.

S1 = V<sub>CC</sub> for V<sub>OL</sub> test.

S1 = GND for V<sub>OH</sub> test.

S1 = V<sub>CC</sub> for High Impedance to active low and active low to High Impedance measurements.

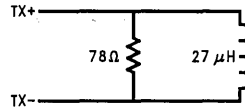
S1 = GND for High Impedance to active high and active high to High Impedance measurements.

### Capacitance T<sub>A</sub> = 25°C, f = 1 MHz

| Symbol           | Parameter          | Typ | Units |
|------------------|--------------------|-----|-------|
| C <sub>IN</sub>  | Input Capacitance  | 7   | pF    |
| C <sub>OUT</sub> | Output Capacitance | 7   | pF    |

### Derating Factor

Output timings are measured with a purely capacitive load for 50 pF. The following correction factor can be used for other loads: C<sub>L</sub> ≥ 50 pF + 0.3 ns/pF.



TL/F/11240-43

**Note:** In the above diagram, the TX<sup>+</sup> and TX<sup>-</sup> signals are taken from the AUI side of the isolation (pulse transformer). The pulse transformer used for all testing is the Pulse Engineering PE64103.

## DP83957 10 Mb/s Repeater Information Base

### General Description

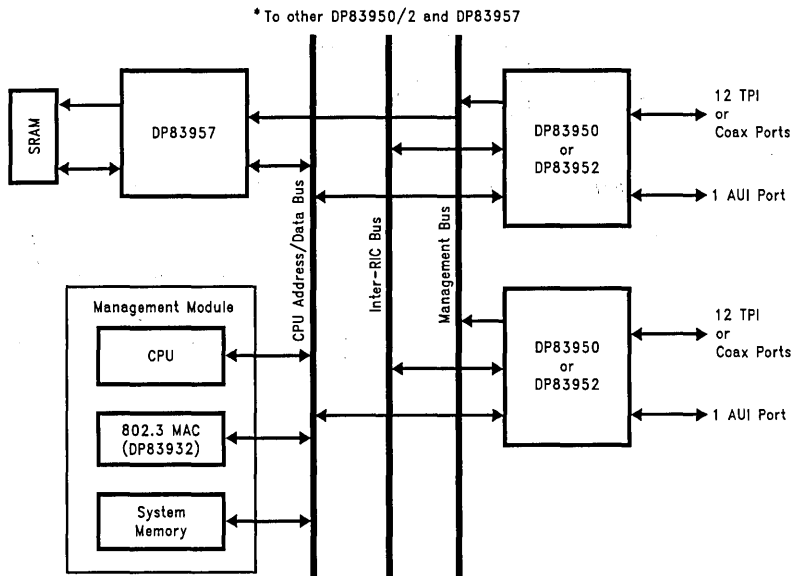
The DP83957 Repeater Information Base "RIB" allows effective implementation of managed multi-port 10 Mb/s ethernet repeater hubs. The device supports all the necessary Port Attributes for the IEEE 802.3k standard (Layer Management for 10 Mb/s Baseband Repeaters). This device is designed to be interfaced with Repeater Interface Controllers (DP83950/2) and gathers attributes on a per-port, per-packet basis.

The DP83957 collects management attributes via the DP83950/2 Management Bus and stores this information in an external SRAM. The DP83957 can be programmed to collect information for up to two DP83950/2 on a shared Management Bus. It can also be used in a dedicated one DP83957 per DP83950/2 architecture. The attributes stored in external SRAM can be easily accessed through the DP83957 by an 8-bit multiplexed Address/Data bus.

### Features

- Fully IEEE 802.3K-July 1992 compatible
- Supports up to two DP83950/2
- Supports generic 1k x 8 or 2k x 8 SRAMs with variable access times of up to 45 ns
- Guarantees at least one attribute access by the micro-processor in the busiest network scenario
- Pipelined architecture for processing back to back frames
- Supports maskable interrupts for status and error reporting on a shared interrupt pin
- Detection and notification of DP83950 Management Bus errors
- Detection and notification of overflow of attribute counters
- 80-pin PQFP package

### 1.0 System Diagram



TL/F/12437-1

## Table of Contents

### 1.0 SYSTEM DIAGRAM

### 2.0 PIN CONNECTION DIAGRAM

### 3.0 PIN DESCRIPTION

- 3.1 CPU Interface
- 3.2 SRAM Interface
- 3.3 Management Bus Interface
- 3.4 Miscellaneous Pins
- 3.5 Pin Type Designation

### 4.0 BLOCK DIAGRAM

### 5.0 FUNCTIONAL DESCRIPTION

- 5.1 SRAM Interface
  - 5.1.1 SRAM Configuration
  - 5.1.2 SRAM Read Operation
  - 5.1.3 SRAM Write Operation
  - 5.1.4 SRAM Memory Map
  - 5.1.5 SRAM Address Format
- 5.2 Management Interface
  - 5.2.1 Management Configuration
  - 5.2.2 Management Bus Interface
  - 5.2.3 Management Bus Processing
  - 5.2.4 Port Attributes
  - 5.2.5 Port Attribute Status
  - 5.2.6 Port Attribute Overflow
- 5.3 CPU Interface
  - 5.3.1 Register Read Operation
  - 5.3.2 Register Write Operation
  - 5.3.3 Interrupts
- 5.4 Registers

### 6.0 REGISTER DESCRIPTIONS

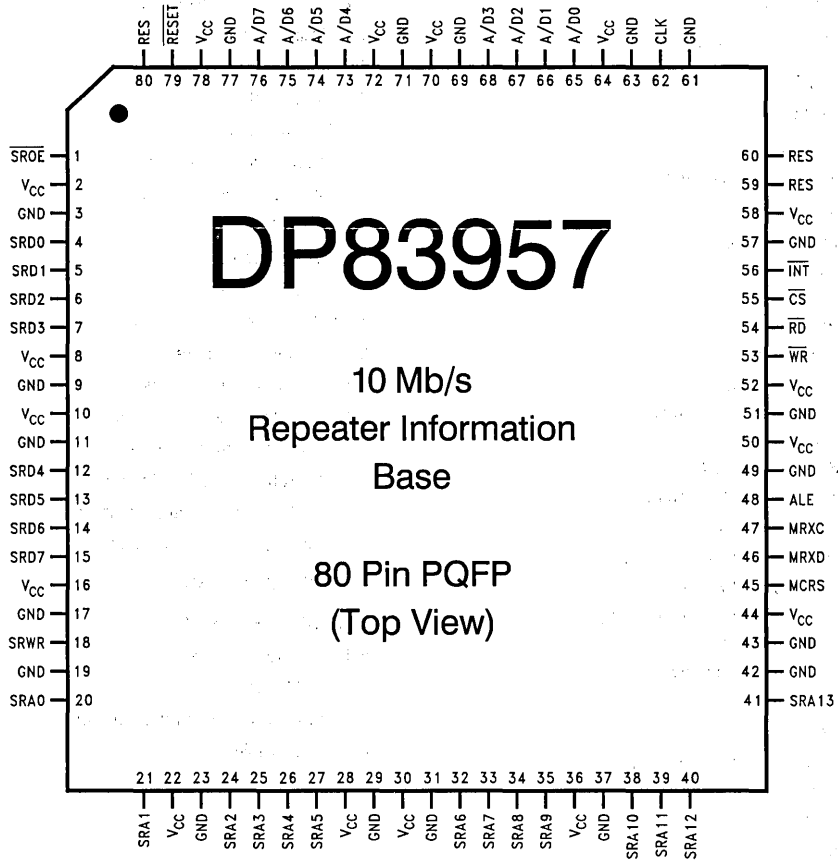
- 6.1 Register Map
- 6.2 Configuration Register 1
- 6.3 Configuration Register 2
- 6.4 Configuration Register 3
- 6.5 Interrupt Mask Register
- 6.6 Interrupt Status Register
- 6.7 Read Data Transfer Registers 1–6
- 6.8 Access Register 1
- 6.9 Access Register 2
- 6.10 Write Data Byte Register
- 6.11 DP83950 ID 1 Register
- 6.12 DP83950 ID 2 Register
- 6.13 Overflow Status 1 Register
- 6.14 Overflow Status 2 Register
- 6.15 DP83957 Revision Register
- 6.16 Port Attribute Status Register 1
- 6.17 Port Attribute Status Register 2

### 7.0 DC SPECIFICATIONS

### 8.0 AC SPECIFICATIONS

- 8.1 CPU Read Timing
- 8.2 CPU Write Timing
- 8.3 Management Bus Interface Timing
- 8.4 Reset Timing

## 2.0 Pin Connection Diagram



TL/F/12437-3



## 3.0 Pin Description

### 3.1 CPU INTERFACE

The CPU interface is a generic 8-bit multiplexed Address/Data bus. The asynchronous interface requires minimum glue logic.

| Signal Name      | Type    | Active | Description                                                                                                                                         |
|------------------|---------|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| $\overline{CS}$  | O/C     | Low    | <b>CHIP SELECT:</b> The Chip Select is generated by the system to select the DP83957 registers. Chip Select must remain valid for the entire cycle. |
| $\overline{RD}$  | O/C     | Low    | <b>READ:</b> Read Data Strobe. The system asserts this pin low to read the DP83957 registers.                                                       |
| $\overline{WR}$  | O/C     | Low    | <b>WRITE:</b> Write Data Strobe. The system asserts this pin low to write to the DP83957 registers.                                                 |
| ALE              | I/C     | High   | <b>ALE:</b> Address Latch Enable. The system needs to drive this to inform the DP83957 that there is a valid address on the Address/Data bus.       |
| $\overline{INT}$ | O/Z/C   | Low    | <b>INTERRUPT:</b> Indicates that an interrupt (if enabled) is pending from one of the sources set in the Interrupt Status Register.                 |
| A/D[7:0]         | O/I/Z/C | —      | <b>ADDRESS/DATA BUS:</b> 8-bit multiplexed CPU address/data bus.                                                                                    |

### 3.2 SRAM INTERFACE

The SRAM interface is used to connect the DP83957 to an external SRAM. The DP83957 supports either a 1k x 8 or 2k x 8-bit SRAM. The SRAM interface can support SRAM access times of 25 ns and 45 ns.

| Signal Name | Type  | Active | Description                                                                                                       |
|-------------|-------|--------|-------------------------------------------------------------------------------------------------------------------|
| SRD[7:0]    | O/I/Z |        | <b>SRAM DATA BUS:</b> This data bus should be connected directly to the external SRAM.                            |
| SRA[13:0]   | O/C   | —      | <b>SRAM ADDRESS BUS:</b> This address bus should be connected directly to a 1k x 8 or 2k x 8 SRAM Address inputs. |
| SRW         | I/C   | —      | <b>SRAM READ-WRITE:</b> A high level signal indicates a Read and a low level signals indicates a Write cycle.     |
| SROE        | I/C   | Low    | <b>SRAM OUTPUT ENABLE:</b> This should be connected directly to the external SRAM output enable.                  |

### 3.0 Pin Description (Continued)

#### 3.3 MANAGEMENT BUS INTERFACE

The Management Bus Interface provides the input to the DP83957 from the DP83950 or the DP83952. The Management Bus connects directly to the DP83950 or DP83952 or through buffers depending on the repeater design (refer to the DP83950 data sheet).

The Management Bus is a NRZ bus that provides the necessary information to the DP83957 so that it can assemble port attributes on a per-port, per-packet basis.

| Signal Name | Type | Active                   | Description                                                                                                                                                                            |
|-------------|------|--------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| MRXC        | I/C  | —                        | <b>MANAGEMENT RECEIVE CLOCK:</b> When asserted this signal provides a clock signal for the MRXD serial data stream. The MRXD signal is changed on the falling edge of this clock.      |
| MRXD        | I/C  | —                        | <b>MANAGEMENT RECEIVE DATA:</b> When asserted this signal provides a serial data stream in NRZ format. The data stream is made up of the packet data and DP83950/2 status information. |
| MCRS        | I/C  | High or Low (Selectable) | <b>MANAGEMENT CARRIER SENSE:</b> When asserted this signal provides an activity framing enable for the serial output data stream (MRXD).                                               |

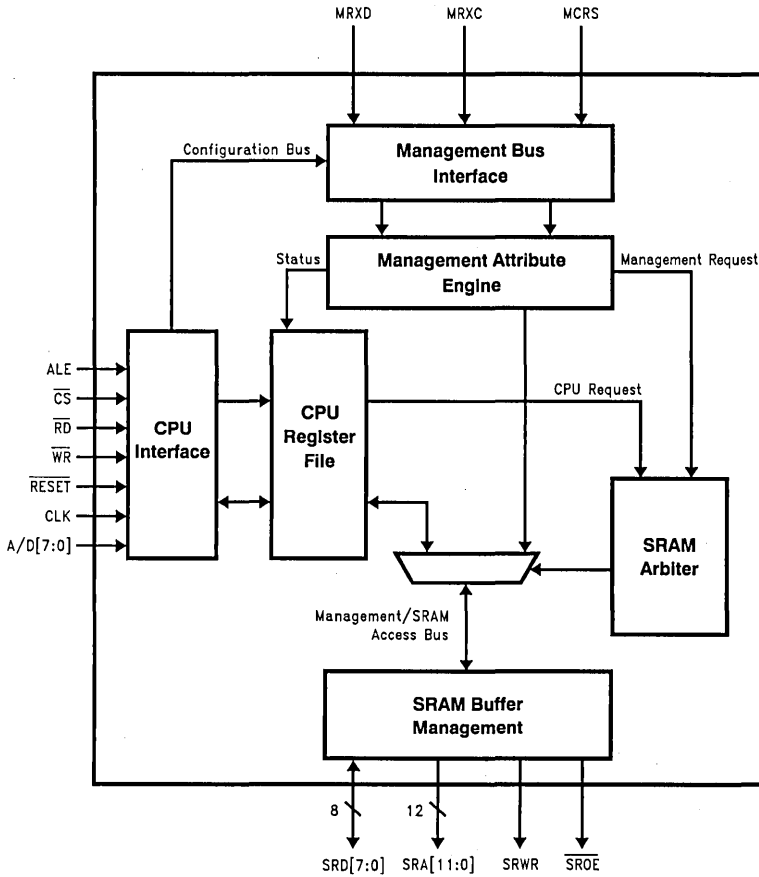
#### 3.4 MISCELLANEOUS PINS

| Signal Name     | Type | Active | Description                                                                                                                                                                       |
|-----------------|------|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| CLK             | I/C  | —      | <b>CLOCK:</b> 40 MHz clock for the internal state machines. This clock does not influence the asynchronous CPU Interface. This clock can be obtained directly from the DP83950/2. |
| RESET           | I/C  | Low    | <b>RESET:</b> The DP83957 is reset when this signal is asserted low. Asserting this signal will cause all the state machines and registers to enter their reset state.            |
| RES             | —    | —      | <b>RESERVED:</b> Leave as No Connect.                                                                                                                                             |
| V <sub>CC</sub> | I    | —      | <b>V<sub>CC</sub>:</b> +5V supply.                                                                                                                                                |
| GND             | I    | —      | <b>GND:</b> Ground return.                                                                                                                                                        |

#### 3.5 PIN TYPE DESIGNATION

| Type  | Description                                          |
|-------|------------------------------------------------------|
| I     | Input Buffer                                         |
| O     | Output                                               |
| Z     | High Impedance State                                 |
| I/O/Z | Bi-directional buffer with high impedance capability |
| C     | CMOS input or output                                 |

### 4.0 Block Diagram



TL/F/12437-2

## 5.0 Functional Description

This section describes the three major blocks of the DP83957. These are the:

- 1) SRAM Interface
- 2) Management Interface
- 3) CPU Interface

### 5.1 SRAM INTERFACE

The DP83957 interfaces directly with standard, off-the-shelf, 1k x 8 or 2k x 8 fast SRAM without any additional buffering. The DP83957 can support access times of 25 ns and 45 ns. If one DP83950/2 is used, then a minimum 1k x 8 SRAM is required.

If two DP83950/2's are used (on a shared management bus), then a minimum 2k x 8 SRAM is required.

The SRAM connected to the DP83957 stores the Management Information Base (MIB) attributes that are extracted from the DP83950's management bus.

The SRAM Interface consists of the following pins:

|           |                      |
|-----------|----------------------|
| SRA[13:0] | 14-Bit Address Lines |
| SRD[7:0]  | 8-Bit Data Bus       |
| SRWR      | Write Enable         |
| SROE      | Output Enable        |

The SRAM interface does not include a chip select output to enable the SRAM, as the SRAM is dedicated to the DP83957. Therefore, the SRAM's chip select input must always be enabled.

#### 5.1.1 SRAM Configuration

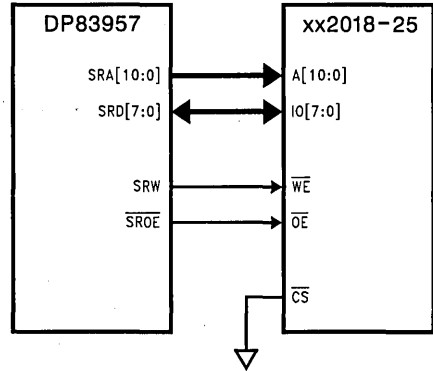
The DP83957 must be configured correctly to allow it to interface with the SRAM. This is achieved through Configuration Registers 1 and 2 (refer to Sections 6.2 and 6.3 for a description of these registers).

The access time of the SRAM can be specified through the SRAM\_ACC bit (D4) of Configuration Register 2.

If bit D4 = 0 then a SRAM access time of 25 ns is selected. If bit D4 = 1 then an access time of between 25 ns and 45 ns is selected.

The size of the SRAM must also be specified through the SIZE bit (D3) of Configuration Register 2. If bit D3 = 0, then a 1k x 8 is selected. If bit D3 = 1, then a 2k x 8 is selected.

The following diagram shows the external SRAM connection to the DP83957.



TL/F/12437-4

External SRAM Connection to DP83957

#### 5.1.2 SRAM Read Operation

A read of the SRAM can occur due to a CPU request for one of the PORT attributes, or due to normal updating of attributes by the DP83957 after receiving the management statistics from the DP83950.

In the latter case, the access to the SRAM is achieved automatically by the DP83957. In the case of a CPU request, the SRAM address must be specified in Access Registers 1 and 2 (refer to Sections 6.8 and 6.9).

The REP\_SEL bit (D4) of Access Register 2, specifies the DP83950 ID, and the PORT\_ID bits (D[3:0]) specify the PORT ID. Together they select the SRAM page. The offset within the page, i.e. the attributes, is specified through Access Register 1. The ACC\_![5:0] bits (D[5:0]) specify the attribute (offset) to be read (Refer to Section 5.1.5, SRAM Address Format).

The DP83957 can read data from the SRAM in variable byte lengths. The number of bytes to be read is specified through bits D[0:2] of Configuration Register 1. The DP83957 can read up to 6 bytes from the SRAM at any one time. The data that is read from the SRAM is placed in Read Data Registers 1-6.

The CPU initiates an SRAM read by setting the ST-RD bit (D4) of Configuration Register 1. The DP83957 resets this bit after the read has completed. It also indicates completion of the SRAM read operation by sending an interrupt (if enabled) to the CPU and setting RD\_COM bit (D4) of the Interrupt Status Register.

## 5.0 Functional Description (Continued)

### 5.1.3 SRAM Write Operation

A write to the SRAM can occur due to a CPU request to one or all of the MIB attributes, or due to normal updating of attributes by the DP83957 after receiving the management status bytes from a DP83950.

In the latter case, the access to the SRAM is done automatically by the DP83957. In the case of a CPU request, the SRAM address must be specified in the same manner as the read operation (refer to Section 5.1.2).

The DP83957 can write data to the SRAM one byte at a time, or it can fill the entire SRAM with the same value. The value to be written into the SRAM is specified in the Write Data Byte register.

The CPU initiates a one-byte SRAM write by setting the ST\_WR bit (D5) in Configuration Register 1.

In order to fill the entire SRAM, both the FILL (D3) and the ST\_WR (D5) bits must be set in Configuration Register 1.

This fill-mode is used to initialize the SRAM to a common value. For example, reset all the attributes to zero.

The DP83957 indicates the completion of the SRAM write operation by sending an interrupt (if enabled) to the CPU and setting the WR\_COM bit (D5) in the Interrupt Status Register.

### 5.1.4 SRAM Memory Map

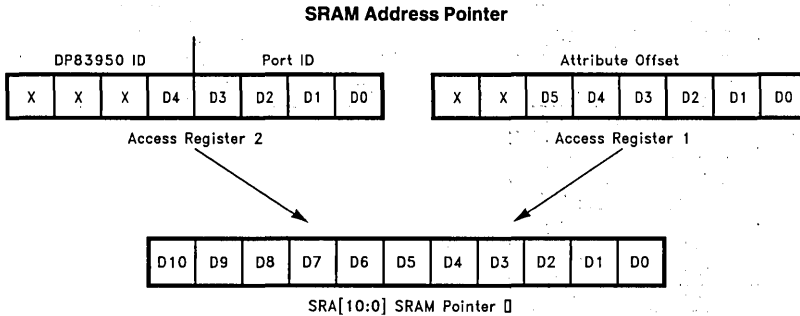
The memory map for the DP83957 consists of two sections. Each section represents one DP83950 device, as shown in the SRAM Memory Map.

For each DP83950, the map is further divided into 13 pages; where each page represents a physical port for the DP83950. The DP83950 ID and Port number selection is programmed into Access 2 Register.

Each port page is further sub-divided into 64 locations. These locations represent the offset of the Port's Attributes. This offset is programmed into Access 1 Register.

### 5.1.5 SRAM Address Format

The SRAM SRA[10:0] address is made up by combining the contents of Access Registers 1 and 2. This is shown in the SRAM Address Pointer figure below.



TL/F/12437-5

## 5.0 Functional Description (Continued)

### SRAM MEMORY MAP

| DP83950 Number  | Port Number | 11-Bit Address |
|-----------------|-------------|----------------|
| DP83950<br>ID 0 | Port 0      | 0x000          |
|                 | Port 1      | 0x040          |
|                 | Port 2      | 0x080          |
|                 | Port 3      | 0x0C0          |
|                 | Port 4      | 0x100          |
|                 | Port 5      | 0x140          |
|                 | Port 6      | 0x180          |
|                 | Port 7      | 0x1C0          |
|                 | Port 8      | 0x200          |
|                 | Port 9      | 0x240          |
|                 | Port 10     | 0x280          |
|                 | Port 11     | 0x2C0          |
|                 | Port 12     | 0x300          |
|                 | Reserved    | 0x340-3FF      |
| DP83950<br>ID 1 | Port 0      | 0x400          |
|                 | Port 1      | 0x440          |
|                 | Port 2      | 0x480          |
|                 | Port 3      | 0x4C0          |
|                 | Port 4      | 0x500          |
|                 | Port 5      | 0x540          |
|                 | Port 6      | 0x580          |
|                 | Port 7      | 0x5C0          |
|                 | Port 8      | 0x600          |
|                 | Port 9      | 0x640          |
|                 | Port 10     | 0x680          |
|                 | Port 11     | 0x6C0          |
|                 | Port 12     | 0x700          |

|                          | Number                    | 6-Bit Address |
|--------------------------|---------------------------|---------------|
| Port N                   | aReadableFrames           | 0x00          |
|                          | aReadableOctets           | 0x04          |
|                          | aFrameCheckSequenceErrors | 0x08          |
|                          | aAlignmentErrors          | 0x0C          |
|                          | aFramesTooLong            | 0x10          |
|                          | aShortEvents              | 0x14          |
|                          | aRunts                    | 0x18          |
|                          | aCollisions               | 0x1C          |
|                          | aLateEvents               | 0x20          |
|                          | aVeryLongEvents           | 0x24          |
|                          | aDataRateMismatches       | 0x28          |
|                          | aLastSourceAddress        | 0x2C          |
|                          | aSourceAddressChanges     | 0x32          |
|                          | Port Attributes Status 1  | 0x36          |
| Port Attributes Status 2 | 0x37                      |               |
| Reserved                 | 0x38-3F                   |               |

TL/F/12437-6

### 5.2 MANAGEMENT INTERFACE

The DP83957 interfaces with either one or two DP83950/2 devices to obtain the 7 bytes of network management statistics appended at the end of each packet on the management bus (refer to DP83950 data sheet).

The management interface consists of the Management Carrier Sense (MCRS), Management Receive Clock (MRXC), and the Management Data (MRXD) signals.

The management interface signals connect directly to the corresponding pins of the DP83950/2.

#### 5.2.1 Management Configuration

The attached DP83950/2 devices can be programmed to have an active high or low Management Carrier Sense (MCRS) signal.

The MCRS\_LEV bit (D0) of Configuration Register 3 is used to select the active level of the MCRS signal from the DP83950.

If bit D0 = 0, then an active low MCRS signal is selected. If bit D0 = 1, then an active high MCRS is selected.

The DP83952 can operate in a Secure or Non-Secure (DP83950) mode. When set in Secure mode, it is possible for the DP83952 to modify the 7 bytes of management.

To cater for this, the DP83957 must be programmed to accept the modified 7 bytes of management status.

The SECURE bit (D7) of Configuration Register 2 is used to specify the operating mode of the DP83952.

If bit D7 = 0, then non-secure mode is selected. If bit D7 = 1, then secure mode is selected.

## 5.0 Functional Description (Continued)

### 5.2.2 Management Bus Interface

The management interface of the DP83957 is responsible for receiving, storing, validating management data, generating all attribute data and updating the attributes stored in the SRAM via the SRAM interface.

The DP83957 receives the per-port statistics information through the management interface (from a DP83950/2) and extracts the attribute information by latching in the last 56 bits (7 bytes) of data. The 7 bytes of data are used to update the appropriate attributes for the specified port.

After the de-assertion of MCRS, the DP83957 generates all attribute information and performs a read-modify-write operation to update the attribute counters stored in the SRAM for the specified DP83950 ID and port ID.

### 5.2.3 Management Bus Processing

The CPU initiates management data bus processing by setting the START bit (D7) in Configuration Register 1.

During the processing of management bus information, one of three error conditions may occur. This error will generate an interrupt to the microprocessor, if the interrupt is enabled.

Three sources of error interrupts that can occur are:

#### Source Address Mismatch (SAM) Interrupt

If the last packet received on a particular port has a Source Address that differs from the aLastSourceAddress attribute for that port, then the SAM bit (D7) will be set in the Interrupt Status Register.

#### No Start Frame Delimiter (SFD\_ERR) Interrupt

If no Start Frame Delimiter (SFD) is detected within the MCRS envelope (i.e. before the end of the packet) on the Management Bus, then the SFD\_ERR bit (D6) will be set in the Interrupt Status Register.

The management status must contain a minimum of 56 bits.

#### Invalid Port ID (IPN) Interrupt

If the received management status bytes contain a valid DP83950 ID, but an invalid port ID (i.e. a Port ID normally greater than 13), then the IPN bit (D3) will be set in the Interrupt Status register.

### 5.2.4 Port Attributes

The DP83957 supports all the Port Attributes that are required to comply with the IEEE 802.3k-1992 Layer Management standard for 10 Mb/s Baseband Repeaters, with the exception of aPortID, aPortAdminState and aAutoPartition attributes.

The aPortID, aPortAdminState and the aAutoPartition attributes are controlled directly by the DP83950.

#### aPortID

Each unique Port ID can be obtained from the DP83950 ID and the individual Port number (0–13) for each DP83950 or it can be obtained indirectly from the DP83957. The Port ID must be kept in software.

#### aPortAdminState

This is used to disable or enable a port such that when it is in disable mode it cannot transmit or receive data. This function can be controlled through the DP83950 DISPT bit (D7) in the Port Real Time Status Register. It is not controlled by the DP83957.

#### aAutoPartitionState

This indicates the state of the auto partition state machine for a particular port (i.e. whether the port is currently partitioned or not). This state is obtained directly from the DP83950 PART bit (D3) in the Port Real Time Status Register. A software routine needs to poll the Port Status Register to keep the counter updated. This state is not controlled by the DP83957.

#### aReadableFrame

This represents the total number of frames received with a valid frame length. The DP83957 increments the counter by one for each frame whose octet count is greater than or equal to the minFrameSize (64 Bytes) and less than or equal to the maxFrameSize (1518 bytes) and for which FCSError and CollisionEvent are NOT asserted. This is a 32-bit counter.

#### aReadableOctets

This counter increments by the number of octets received by the specific port for each frame that has been determined to be a Readable frame. This is a 32-bit counter.

#### aFrameCheckSequenceErrors

This counts the number of frames detected on each port with an invalid frame check sequence. The counter increments by one for each frame with FCSError signal asserted and the Framing Error and CollisionEvent signals de-asserted, and for those frames with a valid FrameSize. This is a 32-bit counter.

## 5.0 Functional Description (Continued)

### aAlignmentError

This counts the number of frames detected on each port with a FCS error and a framing error. The counter increments by one for each frame where the FCSError and FramingError signals are asserted and CollisionEvent de-asserted. In addition to this, the OctetCount must be greater than or equal to the MinFrameSize and less than or equal to the maxFrameSize. This is a 32-bit counter.

### aFramesTooLong

This counts the number of frames that exceed the maxFrameSize. The counter increments by one for each frame where OctetCount is greater than the maxSizeFrame and which does not have an Alignment or FCSError. This is a 32-bit counter.

### aShortEvents

This counts the number of frames that has CarrierEvent with ActivityDuration less than the ShortEventMaxTime. The ShortEventMaxTime is greater than 74 bits but less than 82 bits. This is a 32-bit counter.

### aRuns

This counter increments by one for each CarrierEvent with ActivityDuration greater than the ShortEventMaxTime (74–82 bits) but less than the validPacketMinTime (64 bytes) and with the CollisionEvent signal de-asserted. This is a 32-bit counter.

### aCollisions

This counter increments by one for any CarrierEvent signal asserted on any port in which the CollisionEvent signal on this port is asserted.

**Note:** Since the DP83957 can only keep track of PortN, or the information regarding the repeated packet, this attribute should be kept in the Port-Event Counter of the DP83950/2. This is a 32-bit counter.

### aLateEvents

This counter increments by one for each CarrierEvent in which the collision is detected after the LateEventThreshold (480–565 bit times) has been reached in each frame. This is a 32-bit counter.

### aVeryLongEvents

This counter increments by one for each CarrierEvent whose ActivityDuration is greater than the MAU jabber lock-up protection timer. This is a 32-bit counter.

### aDataRateMismatches

This counter increments by one if the frequency or data rate of the received packet is detectably mismatched from the local transmit frequency. For this counter to increment the following conditions must apply; CollisionEvent is not asserted and the ActivityDuration is greater than the ValidPacketMinTime. This is a 32-bit counter.

### aSourceAddressChange

This counter increments by one each time the Source Address field, of a ReadableFrame, differs from the previous received Source Address field on the same port. This is a 32-bit counter.

### aLastSourceAddress

This attribute stores the SourceAddress value of the last ReadableFrame received by this port. This is a 48-bit buffer.

### 5.2.5 Port Attribute Status

Port Attribute Status Registers 1 and 2 provide a snap-shot of changes to a ports attribute(s) since the last CPU Read access.

These two registers minimize the number of CPU access needed to obtain the Port's attributes stored in the SRAM. They allow the microprocessor to read only those attributes that have been updated since the last CPU access, instead of reading every attribute for a particular port.

The CPU needs to clear the Port Attribute Status bits after a read to these locations to guarantee that it has taken a snap shot of the Port Attribute changes correctly.

Any subsequent updates to any of the attributes by the DP83957 management block will always set the appropriate bits of the Port Attribute Status octets (refer to Sections 6.16 and 6.17).

### 5.2.6 Port Attribute Overflow

The attribute overflow condition is an important feature as it can be used to monitor the frequency of occurrence of a specific attribute. This feature is intended to allow software to set thresholds for an appropriate attribute and monitor the frequency of the overflow condition.

An attribute overflow occurs when the attribute counter rolls over from 0xFF to 0x00. This condition is indicated by the OVFL bit (D2) in the Interrupt Status Register. The attribute for which the counter has overflowed is specified in the Overflow Status Register 2 (refer to Section 6.14 for the bit definitions).

The DP83950 ID and Port ID associated with the overflow condition is specified in Overflow Status Register 1 (refer to Section 6.13).



## 5.0 Functional Description (Continued)

### 5.3 CPU INTERFACE

This interface allows the CPU to read and write to all of the DP83957's registers and to indirectly access the Port attributes stored in the SRAM (via the DP83957). All read and write accesses are byte wide. The DP83957 operates as a slave device and requires a minimum amount of glue logic to interface with the CPU.

All timing requirements as specified in Section 8.0 (AC Timing Conditions) must be satisfied for proper operation.

The CPU interface consists of a multiplexed Address/Data bus (A/D[7:0]), Chip Select ( $\overline{CS}$ ), Read enable ( $\overline{RD}$ ), Write enable ( $\overline{WR}$ ), Address Latch Enable (ALE), and Interrupt ( $\overline{INT}$ ) signals.

#### 5.3.1 Register Read Operation

A read to one of the DP83957 registers is initiated by the CPU logic asserting  $\overline{CS}$ . The address (of the desired register) is driven onto the A/D[7:0] bus by the CPU and must be stable before the falling edge of ALE. After the T5 time has elapsed, the  $\overline{RD}$  signal can be asserted which switches the direction of the A/D[7:0] bus (to output).

The data driven out onto the A/D[7:0] bus by the DP83957 will not be valid until the T7 time has elapsed. At this point, the A/D bus will display the contents of the desired register (refer to Section 8.1, CPU Read Timing Diagrams).

#### 5.3.2 Register Write Operation

A write to one of the DP83957 registers is initiated by the CPU logic asserting  $\overline{CS}$ . The address (of the desired register) is driven onto the A/D[7:0] bus by the CPU and must be stable before the falling edge of ALE. The data that is written to the DP83957 must be stable before the de-assertion (rising edge) of  $\overline{WR}$  (refer to Section 8.2, CPU Write Timing Diagrams).

#### 5.3.3 Interrupts

The  $\overline{INT}$  is an active low signal driven by the DP83957 to indicate that an interrupt has been generated.

The  $\overline{INT}$  signal is shared between the maskable status interrupts and the error reporting interrupts.

The source of the interrupt can be determined by reading the Interrupt Status Register (refer to Section 6.7 for the bit definition).

The CPU clears the generated interrupt by writing a "1" to that bit in the Interrupt Status Register. Writing a "0" to a bit has no effect.

The following events can cause the DP83957 to drive the  $\overline{INT}$  pin low:

- Source Address Mismatch (SAM)
- Management Bus Error (SFD\_ERR or <56 bits)

- SRAM Write complete (WR\_COM)
- SRAM Read Complete (RD\_COM)
- Invalid Port Number Received (IPN)
- Attribute Overflow (OVFL)

Any of the above events can be masked by setting the corresponding bit in the Interrupt Mask Register (refer to Section 6.5 for the bit definition).

The  $\overline{INT}$  output pin can be disabled (TRI-STATE®) by setting the  $\overline{INT}$  bit D6 in Configuration Register 1.

### 5.4 REGISTERS

The DP83957 has 18 registers that are used to control the operation of the DP83957 and obtain the attribute information stored in the SRAM. The following section provides a brief description of these registers. Refer to Section 6.0, DP83957 Register Description, for a more detailed explanation.

#### Configuration Registers 1-3

These registers are used to control and configure the DP83957.

#### Interrupt Mask Register

This register specifies the events that can cause the  $\overline{INT}$  pin to be driven active by the DP83957.

#### Interrupt Status Register

This register specifies the source (event) that caused the  $\overline{INT}$  pin to be driven by the DP83957. The bit(s) can be cleared by writing a "1".

#### Read Data Registers 1-6

When the CPU requests a SRAM read of an attribute, the DP83957 transfers the data obtained from the SRAM into these holding registers.

#### Access Registers 1-2

Access Registers 1 and 2 specify the address for the SRAM read and write operation. The address consists of a page (DP83950/2 ID and Port ID) and offset (Port attribute).

#### Write Data Byte Register

This register contains the value to be written to a SRAM location during a write operation or during a SRAM fill operation.

#### DP83950 or DP83952 ID Registers 1-2

These registers contain the DP83950/2 ID of the DP83950/2's that are dedicated to this DP83957.

#### Overflow Status Registers 1-2

Overflow Status Registers 1 and 2 contain indirectly the DP83950 ID, the actual Port ID, and the attribute whose counter has rolled over from 0xFF to 0x00.

## 6.0 Register Description

The DP83957 has 18 registers which are addressed in a linear fashion. The Reserved Registers should not be accessed. The register map is given in Section 6.1, followed by a detailed description of each register in Sections 6.2 to 6.17.

### 6.1 REGISTER MAP

| Address (Hex) | Register Name              | Access     |
|---------------|----------------------------|------------|
| 00            | Configuration Register 1   | Read/Write |
| 01            | Configuration Register 2   | Read/Write |
| 02            | Configuration Register 3   | Read/Write |
| 03            | Interrupt Mask             | Read/Write |
| 04            | Interrupt Status           | Read/Write |
| 05-0F         | Reserved                   | —          |
| 10            | Read Data Byte 1           | Read       |
| 11            | Read Data Byte 2           | Read       |
| 12            | Read Data Byte 3           | Read       |
| 13            | Read Data Byte 4           | Read       |
| 14            | Read Data Byte 5           | Read       |
| 15            | Read Data Byte 6           | Read       |
| 16-1F         | Reserved                   | —          |
| 20            | Port Access Register 1     | Read/Write |
| 21            | DP83950 Access Register 2  | Read/Write |
| 22            | Write Data Byte            | Read/Write |
| 23-2F         | Reserved                   | —          |
| 30            | DP83950 ID 1               | Read/Write |
| 31            | DP83950 ID 2               | Read/Write |
| 32-3F         | Reserved                   | —          |
| 40            | Overflow Status Register 1 | Read       |
| 41            | Overflow Status Register 2 | Read       |
| 42-4F         | Reserved                   | —          |
| 50            | DP83957 Revision Register  | Read       |
| 51            | Reserved                   | —          |

## 6.0 Register Description (Continued)

### 6.2 CONFIGURATION REGISTER 1

Configuration Register 1 provides the control to the microprocessor to allow the processor to read and write information into the external SRAM.

**Reset State:** 0x00

| Bit   | Bit Name | Access | Bit Description                                                                                                                                                                                                                                |
|-------|----------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D7    | START    | R/W    | Setting this bit to a 1 will enable the DP83957 to start processing Port Attributes.                                                                                                                                                           |
| D6    | INT      | R/W    | This bit enables/disables the Interrupt pin. When disabled the output is in place in TRI-STATE.<br>0: Enable Interrupt<br>1: Disable Interrupt                                                                                                 |
| D5    | ST_WR    | R/W    | By writing a 1 to this bit will allow the CPU to initiate a SRAM write.<br>The DP83957 will reset this bit to 0 when it has completed a SRAM write.                                                                                            |
| D4    | ST_RD    | R/W    | By writing a 1 to this bit will allow the CPU to initiate a SRAM read.<br>The DP83957 will reset this bit to 0 when it has completed a SRAM read.                                                                                              |
| D3    | FILL     | R/W    | Setting this bit to a 1 will fill the entire SRAM with the data pattern contained in the Write Data Byte Register (0x22).                                                                                                                      |
| D2–D0 | RAC[2:0] | R/W    | The CPU must write the number of bytes it needs to read from the SRAM into these 3 bits. The maximum number of bytes that can be read at any one time is 6.<br>The number of bytes read from SRAM is buffered in the Read Data Byte Registers. |

### 6.3 CONFIGURATION REGISTER 2

Configuration Register 2 sets up the DP83957 for different versions of the Repeater Interface Controller (DP83950 or the Secure DP83952) and different external SRAM parameters (i.e. access speeds and SRAM size).

This register must be set up during initialization.

**Reset State:** 0x00

| Bit   | Bit Name | Access | Bit Description                                                                                                                                                                                                                                                                      |
|-------|----------|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D7    | SECURE   | R/W    | This bit selects between a Secure or a Non-Secure Repeater Interface Controller which is connected to the DP83957.<br>0: Selects the Non-Secure Mode<br>1: Selects the Secure Mode<br><b>Note:</b> This is only relevant for the DP83952. For the DP83950 this bit must be set to 0. |
| D6    | CLR_ATR  | R/W    | An attribute will be reset to zero, after the attribute is read from SRAM, if this bit is set to 1.                                                                                                                                                                                  |
| D5    | RES      | R/W    | This bit must always be set to 0.                                                                                                                                                                                                                                                    |
| D4    | SRAM_ACC | R/W    | This bit determines the access speed of the external SRAM.<br>0: Selects an access speed of up to 25 ns<br>1: Selects an access speed of up to 45 ns                                                                                                                                 |
| D3    | SIZE     | R/W    | This bit selects the size of the external SRAM. The size will depend on whether one or two DP83950's are connected to the DP83957.<br>0: Selects a 1024 byte SRAM<br>1: Selects a 2048 byte SRAM                                                                                     |
| D2–D0 | RES      | —      | Reserved                                                                                                                                                                                                                                                                             |

### 6.4 CONFIGURATION REGISTER 3

Configuration Register 3 is used to set-up the Management Interface between the DP83950(s) and the DP83957. The Repeater Interface Controller (DP83950) allows the user to program the Management Carrier Sense (MCRS) signal to be either active high or active low depending on the particular repeater design.

**Reset State:** 0x00

| Bit   | Bit Name | Access | Bit Description                                                                                                     |
|-------|----------|--------|---------------------------------------------------------------------------------------------------------------------|
| D7–D1 | RES      | —      | Reserved.                                                                                                           |
| D0    | MCRS_LEV | —      | This bit selects the active signal level of the Management Carrier Sense signal.<br>0: Active Low<br>1: Active High |

## 6.0 Register Description (Continued)

### 6.5 INTERRUPT MASK REGISTER (0x03)

This register masks the interrupts that can be generated by the Interrupt Status Register. Writing a 1 to a bit disables the corresponding interrupt. During a hardware Reset all interrupts are enabled.

**Reset State:** 0x00

| Bit   | Bit Name | Access | Bit Description                                                                    |
|-------|----------|--------|------------------------------------------------------------------------------------|
| D7    | SAM      | R/W    | 0: Enable Source Address Match Interrupt<br>1: Mask Source Address Match Interrupt |
| D6    | SFD_ERR  | R/W    | 0: Enable Management Bus Error Interrupt<br>1: Mask Management Bus Error Interrupt |
| D5    | WR_COM   | R/W    | 0: Enable SRAM Write complete Interrupt<br>1: Mask SRAM Write complete Interrupt   |
| D4    | RD_COM   | R/W    | 0: Enable SRAM Read complete Interrupt<br>1: Mask SRAM Read complete Interrupt     |
| D3    | IPN      | R/W    | 0: Enable Invalid Port Number Interrupt<br>1: Mask Invalid Port Number Interrupt   |
| D2    | OVRF     | R/W    | 0: Enable Attribute Overflow Interrupt<br>1: Mask Attribute Overflow Interrupt     |
| D2-D0 | RES      | —      | Reserved                                                                           |

### 6.6 INTERRUPT STATUS REGISTER (0x04)

This register indicates the source of an interrupt when the INT pin goes active. Enabling the corresponding bit in the Interrupt Mask Register allows bits in this register to produce an interrupt. When an interrupt occurs, one or more bits in this register are set to a 1.

A specific interrupt is cleared by writing a 1 to the bit that corresponds to the interrupt. A particular interrupt is retained by writing a 0 to it.

**Reset State:** 0x00

| Bit   | Bit Name | Access | Bit Description                                                                                                                                                                                                                                         |
|-------|----------|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D7    | SAM      | R/W    | Indicates that the last packet's Source Address did not match with the ports aLastSourceAddress attribute.                                                                                                                                              |
| D6    | SFD_ERR  | R/W    | Indicates that no SFD was detected within the MCRS envelope. The management status alone must contain 56 bits.                                                                                                                                          |
| D5    | WR_COM   | R/W    | Indicates that the DP83957 has completed a SRAM Write operation.                                                                                                                                                                                        |
| D4    | RD_COM   | R/W    | Indicates that the DP83957 has completed "N" byte reads from SRAM. The result of the transfer is contained in the Read Data Byte Registers.<br><b>Note:</b> "N" corresponds to the value programmed into the RAC[3:0] bits in Configuration Register 1. |
| D3    | IPN      | R/W    | The Management Status received on the management bus contains a valid DP83950 ID but an invalid port number. This will normally occur if the Port number is greater than 13.                                                                            |
| D2    | OVRF     | R/W    | This bit indicates that an attribute overflow occurred for an attribute specified in the Overflow Status Register 2. The DP83950 ID and Port Number is specified in the Overflow Status Register 1.                                                     |
| D1-D0 | RES      | —      | Reserved                                                                                                                                                                                                                                                |

## 6.0 Register Description (Continued)

### 6.7 READ DATA BYTE REGISTER 1–6 (0x10–0x15)

The Read Data Byte Registers are used to buffer the data requested by the CPU from the external SRAM. The CPU can read data from the SRAM in chunks of 1 to 6 bytes, depending on what values are stored in bits RAC[2:0] of Configuration Register 1.

**Reset State:** Undefined

| Bit   | Bit Name     | Access | Bit Description                                                                                                                                                                                                                                                                    |
|-------|--------------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D7–D0 | RD_DATA[7:0] | Read   | These registers hold "N" bytes of data that are read from external SRAM.<br><br>The first byte read back will be placed in Read Data Register 1 (0x10). Therefore, if a 32-bit counter is read, N will be 4 and the least significant byte will be placed in Read Data Register 1. |

### 6.8 ACCESS REGISTER 1 (0x20)

Access Register 1 selects a Port Attribute (offset) for a selected DP83950 ID and port number. The selected DP83950 and port number is specified in Access Register 2. The ACC\_1[5:0] bits provide the 6-bit attribute offset, within a page.

**Reset State:** 0x00

| Bit   | Bit Name   | Access     | Bit Description                                                                                                         |
|-------|------------|------------|-------------------------------------------------------------------------------------------------------------------------|
| D7–D6 | RES        | —          | Reserved                                                                                                                |
| D5–D0 | ACC_1[5:0] | Read/Write | These 6 bits specify the Port Attribute for a selected DP83950 and Port number. refer to the Port Attribute Memory Map. |

### 6.9 ACCESS REGISTER 2 (0x21)

Access Register 2 provides the page select for the Port Attribute Memory Map. The memory map for each DP83950 is broken up into 13 pages, where each page represents a port on the DP83950.

To access a Port Attribute, Access Register 2 must be programmed to select which of the two DP83950's associated with the DP83957 and the specific port (1–13) needs to be examined. Once Access Register 2 is programmed, the CPU can now go and examine or modify a Port Attribute mentioned in Section 5.2.4.

**Reset State:** 0x00

| Bit   | Bit Name  | Access     | Bit Description                                                                                                                                                                                                               |
|-------|-----------|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D7–D5 | RES       | —          | Reserved                                                                                                                                                                                                                      |
| D4    | RIC_SEL   | Read/Write | This bit selects one of the two DP83950's associated with the DP83957.<br>If RIC_SEL is:<br>0: DP83950 ID 1 is selected<br>1: DP83950 ID 2 is selected                                                                        |
| D3–D0 | PORT[3:0] | Read/Write | These bits specify one of the 13 ports on a DP83950 that needs to be processed. The ports are numbered from 1 to 13, where port 1 is the full AUJ port on the DP83950 (refer to the DP83950 data sheet for more information). |

### 6.10 WRITE DATA BYTE REGISTER (0x22)

The Write Data Byte register holds the value that will be written to external SRAM. Depending on the write operation the value can be used to change certain or all bytes of an attribute or fill the entire SRAM with one value.

This register acts as a holding register for the CPU while the DP83957 is busy processing attributes. This allows the CPU to write the value into the register and forget about it, without waiting for the DP83957 to give control to the CPU.

| Bit    | Bit Name  | Access     | Bit Description                                                                              |
|--------|-----------|------------|----------------------------------------------------------------------------------------------|
| D[7:0] | WR_D[7:0] | Read/Write | This holds the data byte value that will be transferred to external SRAM on a write command. |

## 6.0 Register Description (Continued)

### 6.11 DP83950 ID 1 REGISTER (0x30)

The DP83950 ID Registers 1 and 2 are used to associate one or two DP83950's in a multi-DP83950 repeater system, with a DP83957.

The ID address which is placed in these registers must correspond to the value that is placed in the DP83950's Address Register. These registers are located at Page 0 Address 0x17 in DP83950. Refer to Section 8.0 RIC Registers of the DP83950 data sheet for more information.

**Reset State:** 0x00

| Bit    | Bit Name     | Access     | Bit Description                                                             |
|--------|--------------|------------|-----------------------------------------------------------------------------|
| D[7:6] | RES          | —          | Reserved                                                                    |
| D[5:0] | RICID_1[5:0] | Read/Write | This is the ID for the first DP83950 which is associated with this DP83957. |

### 6.12 DP83950 ID 2 REGISTER (0x31)

**Reset State:** 0x00

| Bit    | Bit Name     | Access     | Bit Description                                                              |
|--------|--------------|------------|------------------------------------------------------------------------------|
| D[7:6] | RES          | —          | Reserved                                                                     |
| D[5:0] | RICID_2[5:0] | Read/Write | This is the ID for the second DP83950 which is associated with this DP83957. |

### 6.13 OVERFLOW STATUS 1 REGISTER (0x40)

Overflow Status Registers 1 and 2 are used to indicate a Port Attribute overflow. Overflow Status Register 1 must be used in conjunction with Overflow Register 2 to determine the overflow attribute.

If an overflow interrupt occurs, the CPU must, in the first instance, read Overflow Status Register 1 to determine on which DP83950 (i.e. 1 or 2) and Port number the overflow occurred. Following this, Overflow Register 2 needs to be read to determine the actual Port attribute which caused the interrupt.

**Reset State:** 0x00

| Bit    | Bit Name | Access | Bit Description                                                                                                                                                                                                                             |
|--------|----------|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D[7:5] | RES      | Read   | Reserved                                                                                                                                                                                                                                    |
| D4     | RIC_ID   | Read   | This bit indicates which one of the two DP83950's connected to the DP83957 has an attribute overflow.<br><br>This bit is encoded as follows:<br>0: represents DP83950 ID 1<br>1: represents DP83950 ID 2                                    |
| D[3:0] | PORT_NUM | Read   | These 4 bits indicate which one of the 13 ports of the particular DP83950 identified in bit D4, has an attribute overflow.<br><br>The attribute which has overflowed is recorded in the Overflow Status Register 2 (refer to Section 6.14). |

## 6.0 Register Description (Continued)

### 6.14 OVERFLOW STATUS 2 REGISTER (0x41)

Overflow Status Register 2 indicates which one of the port attributes has rolled over from 0xFF to 0x00.

**Reset State:** 0x3F

| Bit    | Bit Name | Access | Bit Description                                                                                                            |
|--------|----------|--------|----------------------------------------------------------------------------------------------------------------------------|
| D[7:0] | OVF[7:0] | Read   | The byte value represents an encoded value (refer to table below) of the attribute that has rolled over from 0xFF to 0x00. |

#### Overflow Status Encoding

The following table outlines the encoded OVF[7:0] bits for a Port Attribute overflow.

| OVF[7:0] Value | Attribute Description               |
|----------------|-------------------------------------|
| 0x01           | aReadableFrame Attribute            |
| 0x02           | aReadableOctets Attribute           |
| 0x03           | aFrameCheckSequenceErrors Attribute |
| 0x04           | aAlignmentErrors Attribute          |
| 0x05           | aFramesTooLong Attribute            |
| 0x06           | aShortEvents Attribute              |
| 0x07           | aRunts Attribute                    |
| 0x08           | aCollisions Attribute               |
| 0x09           | aLateEvents Attribute               |
| 0x0A           | aVeryLongEvents Attribute           |
| 0x0B           | aDataRateMismatches Attribute       |
| 0x0C           | aSourceAddressChange Attribute      |

### 6.15 DP83957 REVISION REGISTER (0x50)

This register may be used to distinguish between different revisions of the DP83957. If this register is read, it will return a different value for each DP83957 revision (contact National Semiconductor for the latest revision information). A write to this register has no effect upon the contents.

| Bit    | Bit Name | Access | Bit Description                    |
|--------|----------|--------|------------------------------------|
| D[7:2] | REV[7:2] | Read   | Always read back as 0              |
| D1     | REV1     | Read   | Highest bit of the revision number |
| D0     | REV0     | Read   | Lowest bit of the revision number  |

## 6.0 Register Description (Continued)

### 6.16 PORT ATTRIBUTE STATUS REGISTER 1

The Port Attribute Status Registers 1 and 2 are located in the Port Attribute memory map of each port. Port Attribute Registers 1 and 2 are used to indicate if any attributes have changed since the last CPU access to a particular port. These registers therefore provide a status indication of any attribute changes.

The CPU needs to ensure that the port Attributes are cleared after a CPU read to these two registers. This will guarantee that the CPU has taken a snap-shot of the Port Attributes correctly. The Management block of the DP83957 automatically updates the attributes on a port-per-packet basis.

**Reset State:** Undefined (SRAM)

| Bit | Bit Name | Access     | Bit Description                                                                                        |
|-----|----------|------------|--------------------------------------------------------------------------------------------------------|
| D7  | LEVNT    | Read/Write | This status bit indicates whether a port experienced a Late Event.                                     |
| D6  | COL      | Read/Write | This status bit indicates whether a port experienced a Collision.                                      |
| D5  | RUNT     | Read/Write | This status bit indicates whether a port experienced a Runt.                                           |
| D4  | SEVNT    | Read/Write | This status bit indicates whether a port experienced a Short Event.                                    |
| D3  | FTL      | Read/Write | This status bit indicates whether a port experienced a Frame Too Long condition.                       |
| D2  | AERR     | Read/Write | This status bit indicates whether a port experienced an Alignment Error.                               |
| D1  | FCSEERR  | Read/Write | This status bit indicates whether a port experienced a Frame Check Sequence Error on a packet.         |
| D0  | RFO      | Read/Write | This status bit indicates whether a port experienced a Readable Frame or a frame with Readable Octets. |

### 6.17 PORT ATTRIBUTE STATUS REGISTER 2

Port Attribute Status Register 2 is an extension to the Port Attribute Status Register 1 for any port.

**Reset State:** Undefined (SRAM)

| Bit   | Bit Name | Access     | Bit Description                                                                                                                              |
|-------|----------|------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| D7–D3 | RES      | —          | Reserved                                                                                                                                     |
| D2    | SAC      | Read/Write | This status bit indicates whether a port experienced a change of Source Address from the address stored in the aLastSourceAddress attribute. |
| D1    | DRM      | Read/Write | This status bit indicates whether the port experienced a Data Rate Mismatch.                                                                 |
| D0    | VLE      | Read/Write | This status bit indicates whether the port experienced a Very Long Event.                                                                    |



## 7.0 DC Specifications

### ABSOLUTE MAXIMUM RATINGS

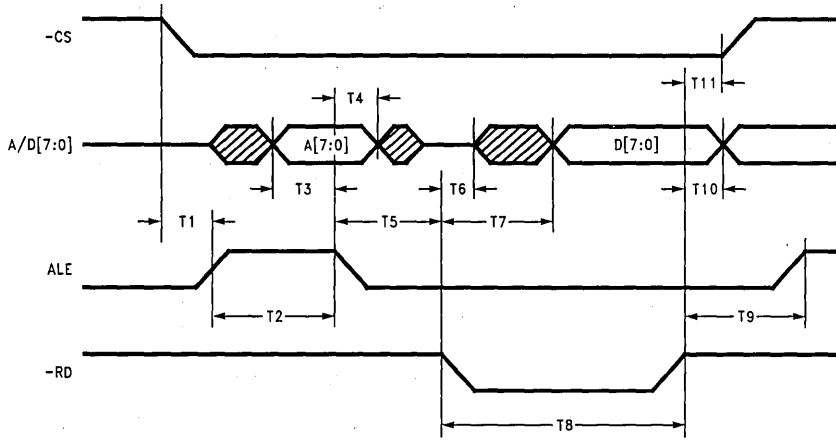
|                                 |                          |                                                   |                 |
|---------------------------------|--------------------------|---------------------------------------------------|-----------------|
| Supply Voltage ( $V_{CC}$ )     | -0.3V to +7V             | Storage Temperature Range ( $T_{STG}$ )           | -65°C to +150°C |
| DC Input Voltage ( $V_{IN}$ )   | -0.5V to $V_{CC} + 0.5V$ | Power Dissipation ( $P_D$ )                       |                 |
| DC Output Voltage ( $V_{OUT}$ ) | -0.5V to $V_{CC} + 0.5V$ | ESD Rating ( $R_{zap} = 1.5k, C_{zap} = 120 pF$ ) | 1.5 kV          |

$T_A = 0^\circ\text{C to } +70^\circ\text{C}$ ,  $V_{CC} = 5V \pm 5\%$ , unless otherwise specified.

| Symbol    | Description                         | Conditions                                                                                                       | Min | Max        | Units   |
|-----------|-------------------------------------|------------------------------------------------------------------------------------------------------------------|-----|------------|---------|
| $V_{OH}$  | Minimum High Level Output Voltage   | $V_{IN} = V_{IH}/V_{IL}$<br>$V_{DD} = 4.5V$<br>$I_{OH} = -1 \text{ mA to } -24 \text{ mA}$                       | 3.7 |            | V       |
| $V_{OL}$  | Maximum Low Level Output Voltage    | $V_{IN} = V_{IH}/V_{IL}$<br>$V_{DD} = 4.5V$<br>$I_{OH} = 20 \mu A$<br>$I_{OH} = 1 \text{ mA to } -12 \text{ mA}$ |     | 0.1<br>0.4 | V<br>V  |
| $V_{IH}$  | Minimum High Level Input Voltage    |                                                                                                                  | 2.0 |            | V       |
| $V_{IL}$  | Maximum Low level Input Voltage     |                                                                                                                  |     | 0.8        | V       |
| $I_{IL}$  | Low Level Input Current             | $V_{IN} = V_{SS}$<br>$V_{DD} = 5.5V$                                                                             |     | -10        | $\mu A$ |
| $I_{IH}$  | High Level Input Current            | $V_{IN} = V_{SS}$<br>$V_{DD} = 5.5V$                                                                             |     | 10         | $\mu A$ |
| $I_{OZL}$ | Low Level TRI-STATE Output Current  | $V_O = V_{SS}$<br>$V_{DD} = 5.5V$                                                                                |     | -10        | $\mu A$ |
| $I_{OZH}$ | High Level TRI-STATE Output Current | $V_{IN} = V_{SS}$<br>$V_{DD} = 5.5V$                                                                             |     | 10         | $\mu A$ |
| $I_{CC}$  | Average Operating Current           |                                                                                                                  |     | 50         | mA      |

# 8.0 AC Specifications

## 8.1 CPU READ TIMING

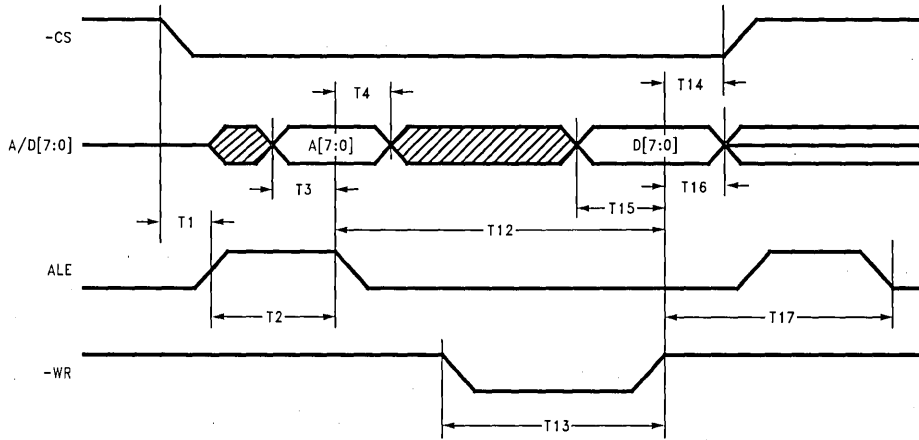


TL/F/12437-7

| Number | Parameter                                   | Min       | Max       | Units |
|--------|---------------------------------------------|-----------|-----------|-------|
| T1     | CS Assertion to ALE High                    | 20        |           | ns    |
| T2     | ALE High Width                              | 20        |           | ns    |
| T3     | Address Setup to ALE                        | 10        |           | ns    |
| T4     | Address Hold from ALE                       | 10        |           | ns    |
| T5     | ALE Deassertion to RD Assertion             | 60        |           | ns    |
| T6     | Read Assertion to Address/Data Bus Enabled  | 0         |           | ns    |
| T7     | Read Data Strobe to Data Valid              |           | bcyc + 22 | ns    |
| T8     | Read Low Width                              | bcyc + 22 |           | ns    |
| T9     | Read Deassertion to ALE Assertion           | 40        |           | ns    |
| T10    | Read Data Strobe to Data TRI-STATE          | 0         |           | ns    |
| T11    | Read Deassertion to Chip Select Deassertion | 0         |           | ns    |

## 8.0 AC Specifications (Continued)

### 8.2 CPU WRITE TIMING

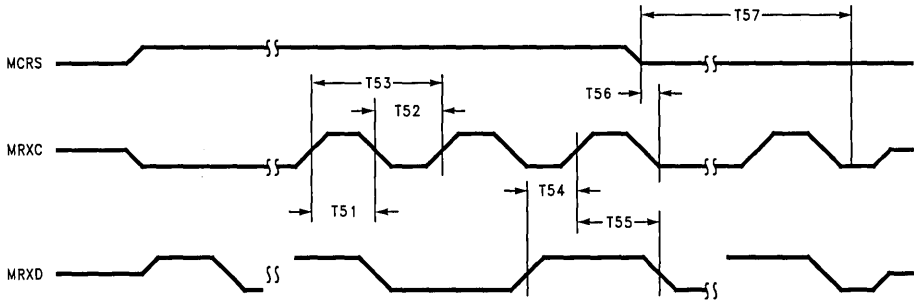


TL/F/12437-8

| Number | Parameter                                       | Min                | Max | Units |
|--------|-------------------------------------------------|--------------------|-----|-------|
| T1     | Chip Selection to ALE High                      | 20                 |     | ns    |
| T2     | ALE High Width                                  | 20                 |     | ns    |
| T3     | Address Setup Time to ALE                       | 10                 |     | ns    |
| T4     | Address Hold Time from ALE                      | 10                 |     | ns    |
| T12    | ALE Deassertion to Write Deassertion            | 60                 |     | ns    |
| T13    | Write Strobe Low Width                          | $\text{bcyc} + 20$ |     | ns    |
| T14    | Write Deassertion to Chip Selection Deassertion | 0                  |     | ns    |
| T15    | Data Setup to Write Data Strobe                 | 10                 |     | ns    |
| T16    | Data Hold from Write Data Strobe                | 10                 |     | ns    |
| T17    | Write Deassertion to ALE Deassertion            | 40                 |     | ns    |

## 8.0 AC Specifications (Continued)

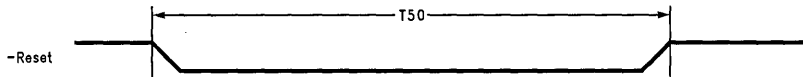
### 8.3 MANAGEMENT BUS INTERFACE TIMING



TL/F/12437-9

| Number | Parameter                                | Min | Max | Units |
|--------|------------------------------------------|-----|-----|-------|
| T51    | MRXC High Time                           | 45  | 55  | ns    |
| T52    | MRXC Low Time                            | 45  | 55  | ns    |
| T53    | MRXC Cycle Time                          | 90  | 110 | ns    |
| T54    | MRXD Setup                               | 40  |     | ns    |
| T55    | MRXD Hold Time                           | 45  |     | ns    |
| T56    | MRXC Low to MCRS Inactive                | -5  | 6   | ns    |
| T57    | Min Number of MRXC's after MCRS Inactive | 5   | 5   | clks  |

### 8.4 RESET TIMING



TL/F/12437-10

| Number | Parameter         | Min | Max | Units |
|--------|-------------------|-----|-----|-------|
| T50    | Reset Pulse Width | 800 | —   | ns    |

# DP83950EB-AT IEEE 802.3 Multi-Port Repeater Evaluation Kit

National Semiconductor  
Application Note 781  
Imad Ayoub



## 1.0 INTRODUCTION

The DP83950EB-AT is a three board kit (Main board, Display Assembly board and Backplane board) that forms an IEEE 802.3 Section 9 Repeater. The Main board has twelve 10Base-T ports and one AUI port and up to four Main boards can be cascaded using the Backplane board to form a larger hub.

The Main board contains the DP83950 Repeater Interface Controller (RIC™), which fits into an IBM PC-AT and compatible computers. The Main board repeats packets, provides management information, updates the Status LEDs, and can be cascaded to other Main boards. The Display Assembly board provides a full set of status LEDs for monitoring the repeater activity, and it provides a breakout to convert the 50-pin connector (with the cable coming from the Main board) to twelve ISO8877 (RJ45) phone connectors. The Backplane board is used for cascading two or more Main boards or to connect to a modified DP839EB-ATS System Oriented Network Interface Controller (SONIC™) Network evaluation board or the new SONIC Network evaluation board, the DP83932EB-AT.

Using the evaluation software the user can read or write to the RIC registers and counters, change the configuration options, enable and disable several features of the RIC, and display graphics of the RIC activities. There are several switches and jumpers on the Main board that configure the board to avoid conflicts with other adapters already installed on the AT bus.

## 2.0 MAIN BOARD OVERVIEW

The block diagram for the Main board is shown in *Figure 1*. The Main board allows the user to exercise all the functions of the RIC while using the twelve 10Base-T ports and the AUI port (the 10Base2 option of the RIC cannot be exercised).

The Main board is designed to perform Mload (refer to the RIC Data Sheet for more information on Mload) through either the Mload Logic (hardware Mload), or through the AT Bus Interface (software Mload). The switches SW1, SW2 are used during the hardware Mload.

The Inter-RIC Arbitration Logic performs the arbitration when there are two or more boards cascaded using the Backplane board. The arbitration is performed when two or more RICs have reception to determine which Main board (i.e., which RIC) is higher in the arbitration chain. The result of the arbitration will be used by the main state diagram of the RIC to determine which port within the RIC has PORT N (or PORT M), as described in the RIC Data Sheet.

The board was designed to allow for choosing between serial and parallel arbitration, and performs the arbitration function accordingly. In the Serial arbitration mode, the RIC logic performs all the arbitration (no additional logic is needed). In the parallel arbitration mode external PALS and logic are required (see Section 2.1.2).

The Inter-RIC BUS Transceivers are used to interface the RIC to the Backplane BUS. The Backplane BUS includes the Inter-RIC signals (IRC, IRD, IRE), the Management signals (MRXC, MRXD, MCRS, PCOMP), the Arbitration and Control signals (ACKI, ACKO, ACTN, ANYXN, COLN) as well as the parallel arbitration vector ARB(3:0). The transceivers are an example of how to perform the transmitting and receiving function over a backplane Bus and interfacing to drive and sense pins on the RIC. The BTL transceivers used allow for long bus applications due to their fast propagation delays and separate bus grounds.

The External Decoder is an example of how the RIC can be configured to run with an external decoder. The Received Manchester data is passed on to the external decoder through the RXM pin, and the decoded NRZ data is sent back from the decoder to the RIC through the Inter-RIC pins IRE, IRC and IRD.

The LED information is sent to the Display board through a driver and a 25-pin ribbon cable.

The 10Base-T Interface includes the buffers, resistors, filters and transformers necessary to interface the RIC ports to the external TP media.

The AUI Interface includes the necessary isolation and resistors to interface to the AUI cable.

In order to enable using up to 16 boards in a PC without using an excessively large address space, all boards can be mapped to a single address block. A separate register is used to enable each individual board. This register is called the Global Register. The Global Register contains other control bits as shown below:

| D6  | D5  | D4  | D3   | D2   | D1   | D0   |
|-----|-----|-----|------|------|------|------|
| EA2 | EA1 | EA0 | RID3 | RID2 | RID1 | RID0 |

Each board in a system is assigned a unique number by setting SW3. When the same number is loaded into the Global Register RID(3:0) as is set by SW3, the RIC Main board is enabled and the RIC's registers can be accessed.

The EA(2:0) bits are used to perform various functions on the selected board in the following manner:

| EA2 | EA1 | EA0 | Function                      |
|-----|-----|-----|-------------------------------|
| 0   | 0   | 0   | Normal Operation              |
| 0   | 0   | 1   | Issue CDEC to All Boards      |
| 0   | 1   | 0   | Issue CDEC to Selected Board  |
| 0   | 1   | 1   | Issue MLOAD to All Boards     |
| 1   | 0   | 0   | Issue MLOAD to Selected Board |

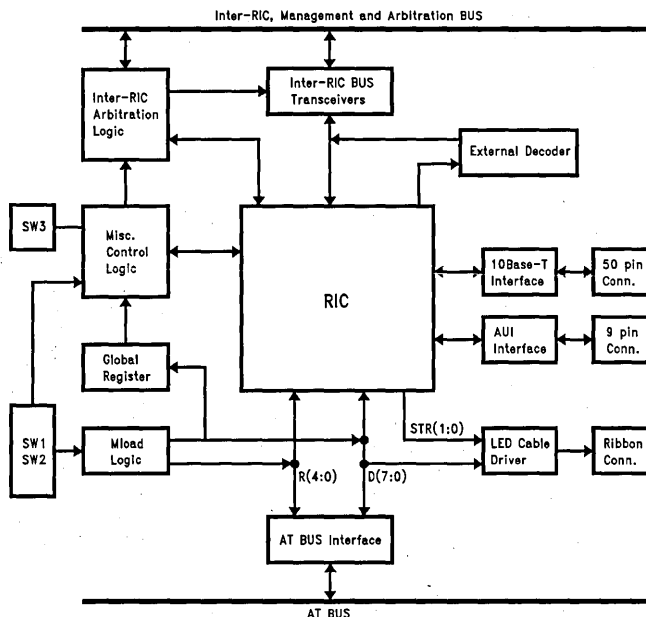


FIGURE 1. Main Board Block Diagram

TL/F/11230-1

All 32 RIC registers and the Global register are IO mapped and can be relocated by setting BA(1:0) in SW1 as follows:

| BA1 | BA0 | RIC Registers | Global Register |
|-----|-----|---------------|-----------------|
| 0   | 0   | 100 h–11F h   | 200 h           |
| 0   | 1   | 120 h–13F h   | 220 h           |
| 1   | 0   | 140 h–15F h   | 240 h           |
| 1   | 1   | 160 h–17F h   | 260 h           |

## 2.1 Detailed Description

### 2.1.1 Mload and AT Bus Interface

To perform the Mload pin configuration the board has the capability to load the D(7:0) and R(4:0) pins by either software or hardware.

Hardware Mload is done by the Mload Logic, which interfaces to pins D(7:0) and R(4:0) on the RIC to configure the RIC upon power up. Switches SW1 and SW2 allow for hardware setting of the Mload pin configuration. An RC network is used to provide a pulse (RSTB) at power up which will be used by a PAL (U41) to assert the Mload signal to the RIC. The PAL is needed to control the enables for the buffers for choosing between the hardware and software Mload.

Software Mload is implemented by passing the D(7:0) and R(4:0) signals from the PC-AT bus through the AT Bus Interface and onto the RIC pins.

When a Global Register write operation is performed by the PC-AT, it is written to a flip-flop (U44) which passes, first, the bits RID(3:0) to a comparator (U38), second, the bits EA(3:0) to the one of the control PALs. The comparator asserts a "RICHIT" if the Global Register RID matches the board number. This will enable the control PALs to perform the operation required by the Global register.

The ELI and RTI pins from the RIC can be passed onto the AT BUS to one of four interrupt request lines on the AT BUS (IRQ(15), IRQ(12), IRQ(11) or IRQ(10)) by selecting the appropriate jumper settings (JB1 and JB2, refer to schematic). Three PALs (U36, U41, U43) are used to control the AT Bus Interface for software Mload and register Read/Write, and to enable the various buffers required for hardware Mload and the Global register decode. The PAL equations listings for all the PALs are included in Section 5.0 of this document.

U43 decodes the AT BUS address bits SA(9:0) and BA(1:0) to determine if the software operation is addressed to this board. A Global hit (Ghit) is asserted when a match occurs with the Main board's global address. A Base hit (Bhit) is asserted when a match occurs with a RIC register.

The AT BUS signals IOW, IOR, and AEN, and the Global register bits EA(2:0) are decoded by the PALs U36 and U41, resulting in the various control signals for the Mload buffers, the Read, Write and CDEC signals, the CHRDY signal to the PC-AT BUS, and the receive enable signals for the Inter-RIC and the management BTL transceivers (U4, U5).

### 2.1.2 Inter-RIC Arbitration Logic

Since there is no central arbiter, each Main board using the Inter-RIC BUS needs a way to tell if it owns the bus. This implementation uses one of two methods: serial or parallel arbitration (by setting JB3, refer to schematic).

In the serial arbitration method the RIC signals ACKI and ACKO are passed to and from the Backplane BUS directly (as SACKI and SACKO) without further arbitration. Therefore the serial arbitration is done by the RIC logic itself. A high level on ACKI tells the RIC that it can take the bus.

Therefore, the physical position of the board controls its priority in the chain. To participate properly in the chain the RIC will pass a high ACKI to ACKO if it does not want to transmit, but will force ACKO low if it does wish to transmit. In addition, ACKO needs to be held low if a low ACKI is seen, in order to tell the boards further down in the chain that they cannot take the bus.

The drawback for serial arbitration is that it requires all the Main boards to be inserted at all times, otherwise the SACKI/SACKO chain will be broken. The Backplane board is equipped with a jumper to connect these signals to continue the arbitration chain.

In the Parallel arbitration mode the SACKI/SACKO signals are controlled by two PALs according to the ACKI, ARBWIN and ENARB state machines shown in Figures 2, 3 and 4. A priority number is assigned via a set of dip switches SEL(3:0), where a higher number corresponds to a higher priority. Each board that wants to use the bus asserts this

vector onto the bus. The bus is "wired-OR", so that only the board with the highest priority will see its own vector reflected back from the bus.

By monitoring the RIC signals and the arbitration vector, the parallel logic controls ACKI, so that normal functionality is maintained and the board's priority is independent of its physical location.

The arbitration takes place whenever a RIC is trying to take the bus, i.e., ACTND is asserted, or when the RIC is experiencing a transmit collision, i.e., ANYXND is asserted (Figure 2). When either the ACTND or ANYXND are asserted the ENARB signal (Figure 4) is asserted enabling the arbitration BTL transceiver (U2).

A counter state machine is used in another PAL (U33) to give some delay for the arbitration to be completed, and asserts the arbitration done signal ARBDONE (Figure 3). Based on the ACKI state machine the ACKIRIC signal is asserted to the RIC.

**ACKI STATE MACHINE**

Inputs: ACTND~, ANYXND~, ARBDONE, ARBWIN  
Output: ACKI

~ = Active Logic Low

! = Inactive

eg: !ACTNB~ = Inactive or Logic High

!ARBWIN = Inactive or Logic Low

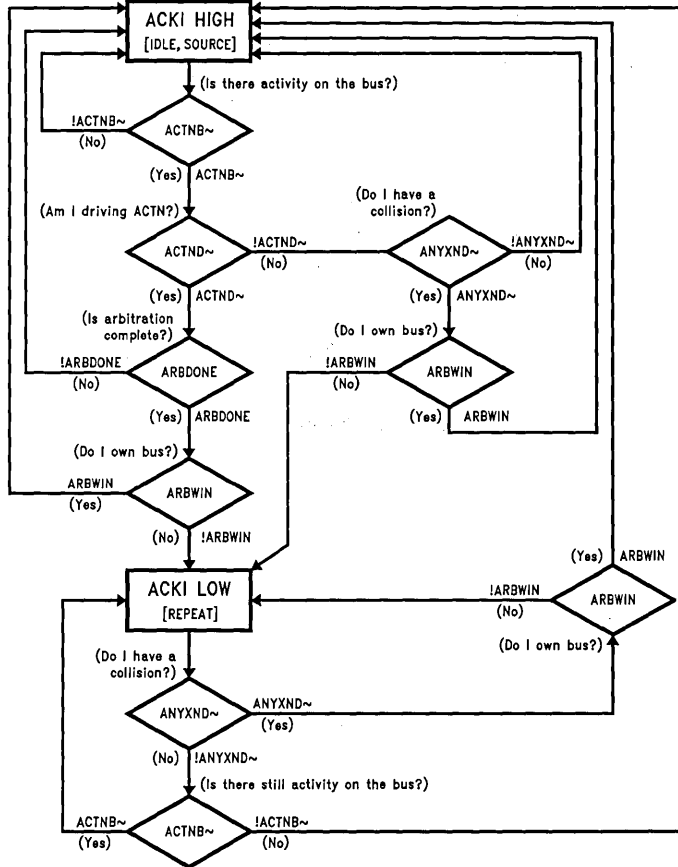


FIGURE 2. ACKI State Machine

TL/F/11230-2

**ARBWIN STATE MACHINE**

Inputs: ANYXND~, ENARB, ARBDONE, BUSWIN~

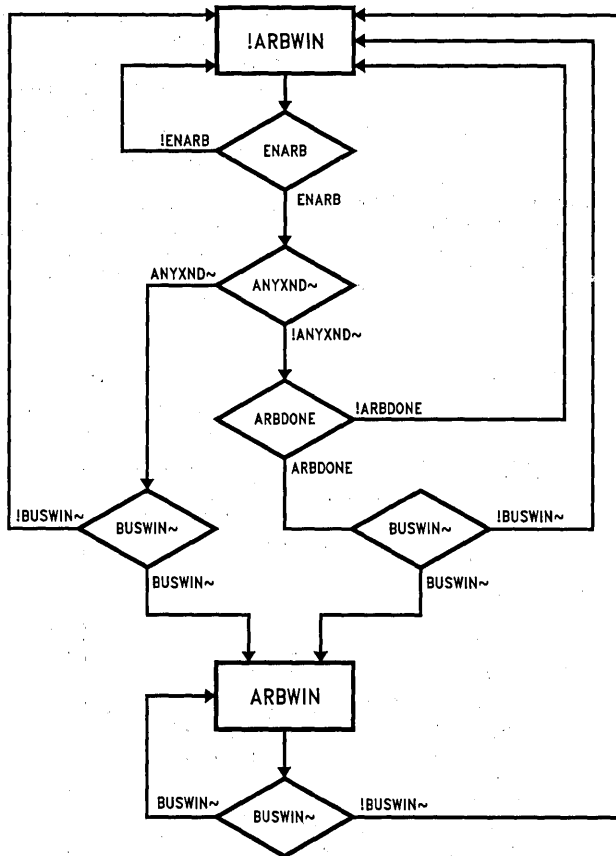
Output: ARBWIN

~ = Active Logic Low

! = Inactive

eg: !ANYXND~ = Inactive or Logic High

ANYXND = Active or Logic Low



**FIGURE 3. ARBWIN State Machine**

TL/F/11230-3



**ENARB STATE MACHINE**

Inputs: ACTND~, ANYXND~

Output: ENARB

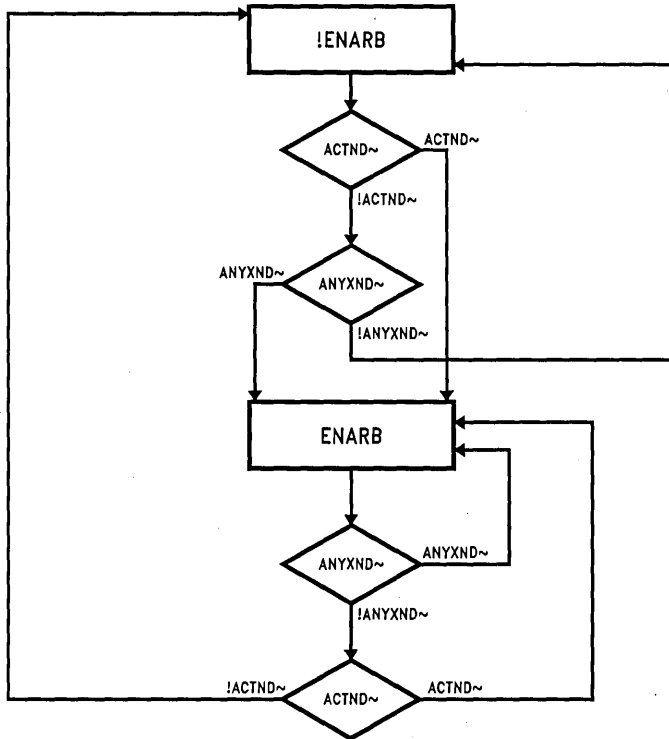
ENARB = ACTND~ # ANYXND~

~ = Active Logic Low

! = Inactive

eg: !ACTND~ = Inactive or Logic High

ACTND~ = Active or Logic Low



**FIGURE 4. ENARB State Machine**

TL/F/11230-4

### 2.1.3 Inter-RIC Bus Transceivers

To form a 10Base-T HUB with more than 12 Twisted Pair ports, up to four Main boards can be cascaded using the Backplane board (up to 16 Main boards can be cascaded with an extended Backplane board). The Backplane board can also be used to pass management information, as specified in the Hub management specification, from the management bus of the RIC to a modified DP839EB-ATS SONIC network evaluation board or the new SONIC Network evaluation board, the DP83932EB-AT.

To assure good speed and signal quality over the backplane bus four BTL (Turbo Transceivers) are used (U2, U3, U4, U5). The required BTL terminations are done on the Backplane board. Tying all the Ground pins of the BTLs together is not the optimum way to use these transceivers, however, it was necessary to assure proper operation when the Main board is in stand alone mode and the backplane board is not inserted. In a typical application where the Backplane BUS is always terminated these grounds would not be grounded together.

High level of assertion for the bidirectional "wired-OR" signals (ACTN, ANYXN, COLN, IRE, MCRS) of the RIC is required for proper operation with the inverting BTL transceivers. This makes these signals asserted low on the BUS side of the transceivers.

The parallel arbitration vectors ARBI(3:0) and ARBO(3:0) are transmitted and received over the BUS through one BTL Transceiver (U2). The receive enable for U2 is controlled by the stand alone (SLN) bit set during Mload. The drive enable for U2 is controlled by the enable arbitration (ENARB) signal from the arbitration PALs.

Another PAL (U3) transmits and receives the ACTN, ANYXN, and PCOMP signals onto the Backplane BUS. On the BUS side of U3, ACTN and ANYXN are bidirectional signals. They are asserted when any RIC asserts its ACTNd or ANYXNd signals. On the RIC side of U3, ACTN is split into ACTNd and ACTNs, and ANYXN is split into ANYXNd and ANYXNs. PCOMP is a unidirectional signal that is asserted onto the BUS by a separate controller board that can gather management statistics (or a modified DP839EB-ATS SONIC-AT board). The Drive enable for U3 is always enabled, allowing the ACTNd and ANYXNd signals to assert

the BUS ACTN and ANYXN signals directly. The Receive enable is disabled only when the Main board is in the stand alone mode (i.e., there are no other Main boards in the HUB).

The RC network included on the ANYXNs signal is recommended (see Application Note #671 in the Interface Data Book for design details using BTL Transceivers).

A third PAL (U4) is used to transmit and receive the IRC, IRD, IRE and COLN signals. The COLN signal, which signals a receive collision, has no significance in the TP media. It is included here for completeness. These signals are bidirectional, however they are unidirectional at any one time. When the RIC is the receiving RIC, it asserts the packet enable signal PKEN signal which is used as the drive enable for U4. PKEN will be asserted as long as the RIC is the receiving RIC. The receive enable ENPKEN is asserted when the Main board is not in stand alone mode, and the PKEN signal is not asserted.

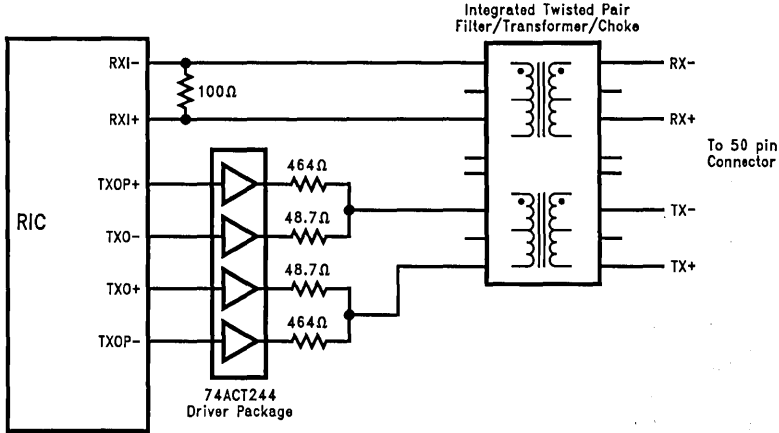
A fourth PAL (U5) is used to transmit the management bus signals MRXC, MCRS and MRXD. The MRXC and MRXD signals are unidirectional RIC output signals, while the MCRS is a bidirectional. The RIC senses the MCRS signal while the RIC is not the receiving RIC to assure the Inter-frame gap limit set in the RIC Inter-Frame Gap Threshold Select Register is not violated before sending another packet onto the Management BUS. The Management BUS information can be received by a SONIC, connected to the management bus. When the RIC is the receiving RIC, it asserts the management enable signal MEN, which is used as the drive enable for U5. MEN will be asserted as long as the RIC is the receiving RIC. The receive enable ENMEN is asserted when the Main board is not in stand alone mode, and the MEN signal is not asserted.

### 2.1.4 External Decoder

The RXM External pin decoder allows using the RIC with an external decoder. The RXM pin outputs the received Manchester Data from the RIC. This data is sent to the DP83910A decoder, which is decoded and passed onto the IRD, IRC and IRE BUS signals through a buffer back to the RIC. The buffer is enabled by PKEN, and a jumper is used to disable the buffer when using the internal decoder mode.

**2.1.5 TP Interface**

The interface is shown for one port in *Figure 5* below:



**FIGURE 5. TP Interface**

TL/F/11230-5

To drive the transmitted signal through 100 meters of Twisted Pair cable, the RIC requires external buffers. The resistor network on the transmit path shows the values used on the Main board. A more optimized network, which allows for better amplitude control is described in the TP Parametrics Evaluation document. The Filter/Transformer/Choke package used here is the PE65431 from Pulse Engineering. Other packages have been evaluated, and those results are described in the TP Parametrics Evaluation Document.

**2.1.6 AUI Interface**

The AUI includes the proper terminations and pulldowns, and the isolation transformer. A 9-pin connector is used instead of the standard 15-pin AUI connector (due to space limitations). A 9-pin to 15-pin special adapter cable is used to attach to the MAU.

**3.0 DISPLAY BOARD DESCRIPTION**

The Display board allows for the display of Maximum mode or Minimum mode LED configurations. The LED display address and data information RD(7:0), and the strobe signals STR(1:0) signals are received from the Main Board through the 25-pin ribbon cable and driver. The data is driven to two arrays of addressable latches and one flip-flop.

In the Maximum LED display mode all 66 LEDs are functional. Five sets of Latches are used and are arranged into five sets, with two latches per set. Each set controls one of the following groups of LEDs: Receive (REC), Collision (COL), Partition (PART), Good Link (GDLINK), and Bad Polarity (BDPOL).

The address bits for the latches are obtained from the RD(7:5) signals. On the top half of the array address 0 corresponds to the "any port", address 1 corresponds to the AUI port, and addresses 2 to 7 corresponds to ports 2 through 7. The top array is enabled by the STR0 signal. The bottom half is enabled by the STR1 signal, and addresses 0 through 5 correspond to ports 8 through 13.

The data is obtained from the RD(4:0) signals as follows: RD(0) for LINK, RD(1) for Collision, RD(2) for Receive, RD(3) for Partition, and RD(4) for Polarity.

In the Minimum LED display mode four LEDs are displayed: Any port collision (ACOL), Any port reception (AREC), Any port jabbering (JAB) and Any port partitioned (APRT), which indicate any activity on any of the RIC ports. In this mode the flip-flop ('ALS374) should be inserted into the socket (U6), which is left blank (default for the Maximum display Mode). The flip-flop passes the four LED signals to port 13, and STR0 is used as the clock.

**4.0 BACKPLANE BOARD DESCRIPTION**

This board forms the Backplane BUS for the HUB. There are four types of signals that are passed on this BUS.

1. The Inter-RIC BUS signals: IRE, IRC, IRE. These signals are passed from the Receiving RIC to all the other RICs in the HUB. They can also be used by a modified DP839EB-ATS SONIC board to allow the SONIC to transmit to the network through all the RICs on the HUB.
2. The Arbitration and Control signals: SACK1, SACKO, ACTN, ANYXN, COLN. These signals are passed between all the RICs on the HUB to assure the proper operation of the HUB per the IEEE802.3 state diagrams. When a board is inserted into a slot on the Backplane board, a jumper is removed to allow for the SACK1-SACKO signals to pass to and be asserted by the Main board. If there is no Main board inserted in a slot, that jumper should be inserted to short SACK1 to SACKO, in order to complete the arbitration chain.
3. The Management BUS signals: MRXC, MRXD, MCRS, PCOMP. The MRXC, MRXD and MCRS signals are used to pass the management information from the receiving RIC to a SONIC board. PCOMP is a unidirectional signal that is asserted onto the BUS by a separate controller board that can gather management statistics to compress the data portion of the management information. The MCRS signal is also used as an input to the RIC as described in Section 2.1.3).
4. The Parallel arbitration vector, ARB(3:0), required for parallel arbitration (as described in Section 2.2.2).

Each of the BUS lines is terminated by approximately 20Ω (two 39Ω resistors in parallel).

## 5.0 PAL LISTINGS

```
U43 device 'p16L8';module RIC_DEC
title      'decode AT addresses'
```

```
"inputs
```

```
sa0      pin 11;
sa1      pin 9;
sa2      pin 8;
sa3      pin 7;
sa4      pin 6;
sa5      pin 5;
sa6      pin 4;
sa7      pin 3;
sa8      pin 2;
sa9      pin 1;
ba0      pin 16;
ba1      pin 17;
aen      pin 18;
iow      pin 13;
ior      pin 14;
```

```
"outputs
```

```
ghit     pin 19;
bhit     pin 12;
io       pin 15;
base0 =  !aen & !sa9 & sa8 & !sa7 & !ba1 & !ba0;
base1 =  !aen & !sa9 & sa8 & !sa7 & !ba1 & ba0;
base2 =  !aen & !sa9 & sa8 & !sa7 & ba1 & !ba0;
base3 =  !aen & !sa9 & sa8 & !sa7 & ba1 & ba0;

low      =  !aen & sa9 & !sa8 & !sa7 & !sa4 & !sa3 & !sa2 & !sa1
          & !sa0;
```

```
equations
```

```
!ghit    =  low & !ba1 & !ba0 & !sa6 & !sa5
          # low & !ba1 & ba0 & !sa6 & sa5
          # low & ba1 & !ba0 & sa6 & !sa5
          # low & ba1 & ba0 & sa6 & sa5;

!bhit    =  !sa6 & !sa5 & base0
          # !sa6 & sa5 & base1
          # sa6 & !sa5 & base2
          # sa6 & sa5 & base3;

!io      =  !iow # !ior;
```

```
END RIC_DEC;
```

```
U36 device 'p20L8';ric_ctrl1
title 'ric control and some AT interface'
```

```
"inputs
```

```
  mloadly pin 23;
  ior      pin 14;
  iow      pin 13;
  rstdrv   pin 11;
  bufen    pin 10;
  rdy      pin 9;
  ea2      pin 8;
  ea1      pin 7;
  ea0      pin 6;
  richit   pin 5;
  ghit     pin 4;
  bhit     pin 3;
  io       pin 1;
```

```
"outputs
```

```
  chrdy    pin 21;
  ireg     pin 20;
  rd       pin 19;
  wr       pin 18;
  iorb     pin 17;
  dens     pin 16;
  cdec     pin 15;
  den      pin 22;

  norm     = !ea2 & !ea1 & !ea0;
```

```
equations
```

```
  !chrdy   = 1;
  enable chrdy = !bhit & richit & rdy & !io;
  !dens    = !io & (!ghit # (!bhit & richit));
  !ireg    = !ghit & !iow;
  !rd      = richit & norm & !bhit & !ior;
  !wr      = richit & norm & !bhit & !iow;
  !iorb    = !ior;
  !cdec    = !ea2 & !ea1 & ea0 & !ghit & !ior
            # !ea2 & ea1 & !ea0 & richit & !ghit & !ior;
  !den     = richit & !bhit & !bufen & norm & !io;

  end ric_ctrl1;
```

TL/F/11230-7

```
U41 device 'p20L8';ric_ctrl2;  
title 'ric control 2a'
```

```
"inputs
```

```
    mloaddly  pin 23;  
    ior       pin 14;  
    iow       pin 13;  
    rstdrv    pin 11;  
    bufen     pin 10;  
    rdy       pin 9;  
    ea2       pin 8;  
    eal       pin 7;  
    ea0       pin 6;  
    richit    pin 5;  
    ghit      pin 4;  
    bhit      pin 3;  
    sln       pin 2;  
    pken      pin 1;  
    rstb      pin 16;  
    men       pin 17;
```

```
"outputs
```

```
    swen      pin 21;  
    raen      pin 20;  
    rst       pin 19;  
    mload     pin 18;  
    enpken    pin 22;  
    enmen     pin 15;
```

```
equations
```

```
    !swen     = !mload & !mloaddly;  
    !raen     = mload & mloaddly;  
    !rst      = rstdrv # !rstb;  
    !mload    = rstdrv  
               # ea2 & !eal & !ea0 & richit  
               # !ea2 & eal & ea0  
               # !rstb;  
    !enpken   = !sln & !pken;  
    !enmen    = !sln & !men;  
  
end ric_ctrl2;
```

```
U33 device 'p20v8r';module ric_ack
title 'ric arb state machine
```

```
"inputs
```

```
    enarb    pin 14;
    unused_1 pin 11;
    psel     pin 23;
    sln      pin 10;
    sacki    pin 9;
    actn_s   pin 8;
    actn_d   pin 7;
    anyxn_d  pin 6;
    unused_3 pin 5;
    unused_4 pin 4;
    unused_5 pin 3;
    match    pin 2;
    clk      pin 1;
```

```
"outputs
```

```
    ackiric  pin 15;
    arbwin   pin 18;
    arbdone  pin 20;
    q1       pin 21;
    q0       pin 22;
    Q20M     PIN 17;
```

```
"counter states
```

```
    s0 = ^b00;
    s1 = ^b01;
    s2 = ^b10;
    s3 = ^b11;
```

```
"counter modes
```

```
    mode = [enarb];
    count = [1];
    clear = [0];
```

```
state_diagram [q1,q0]
```

```
    state s0: case (mode == clear): s0;
                (mode == count): s1;
            endcase;
```

```
    state s1: case (mode == clear): s0;
                (mode == count): s2;
            endcase;
```

```
    state s2: case (mode == clear): s0;
                (mode == count): s3;
            endcase;
```

```
TU/F/11230-9
```

```
state s3: case (mode == clear): s0;
              (mode == count): s3;
endcase;

equations

arbdone = q1 & q0;

arbwin = (!anyxn_d # arbdone) & match & enarb;

!ackiric = !sln & (psel & (ackiric & (actn_s & actn_d & arbdone &
!arbwin # actn_s & !actn_d & anyxn_d & !arbwin)
# !ackiric & (anyxn_d & !arbwin # !anyxn_d & actn_s))
# !psel & !sacki);

!Q20M := Q20M;

end ric_ack;
```

TL/F/11230-10



```
U1 device 'p2018';module ric_arb
title 'arbitration pal
```

```
"inputs
```

```
    arbi0      pin 1;
    arbi1      pin 2;
    arbi2      pin 3;
    arbi3      pin 4;
    sel0       pin 5;
    sel1       pin 6;
    sel2       pin 7;
    sel3       pin 8;
    anyxn_d    pin 9;
    actn_d     pin 10;
```

```
"outputs
```

```
    match      pin 17;
    enarb      pin 22;
    arbo0      pin 18;
    arbo1      pin 19;
    arbo2      pin 20;
    arbo3      pin 21;
```

```
equations
```

```
    enarb      = actn_d # anyxn_d;

    match      = (!arbi3 # sel3) & (!arbi2 # sel2)
                 & (!arbi1 # sel1) & (!arbi0 # sel0);

    arbo3      = sel3;

    arbo2      = sel2 & (!arbi3 # sel3);

    arbo1      = sel1 & (!arbi3 # sel3) & (!arbi2 # sel2);

    arbo0      = sel0 & (!arbi3 # sel3) & (!arbi2 # sel2)
                 & (!arbi1 # sel1);
```

```
end ric_arb;
```

TL/F/11230-11

# RIC™-SONIC™ Interface

National Semiconductor  
Application Note 782  
Imad Ayoub



## INTRODUCTION

This document describes how the DP83950 Repeater Interface Controller (RIC) can be interfaced to a System Oriented Network Interface Controller (SONIC) controller. The emphasis in this note is on the hardware interface between the RIC and the SONIC. The software implementation of the Hub management protocols such as SNMP and CMIP are not discussed in this note, since each system's implementation would be different depending upon the processor used and the number of RICs and SONICs employed in the Hub. A description of the extra logic necessary to interface the RIC to a NIC (DP8390) is included for reference. And last, in order to provide a simple and fast solution for evaluating the RIC's management bus interface to the SONIC, a description of a simple way to hook the SONIC DP839EB-ATS evaluation board to the RIC's management Bus is included.

## RIC-SONIC INTERFACE

The RIC transmits over the management bus every packet that is received from any of the ports (refer to the RIC datasheet for details). The management bus packet is different from the packets transmitted to/from the ports. First, the preamble on this bus is always five bits (01011). Second, at the end of the packet, after the CRC pattern, seven bytes of management status are appended to the packet by the RIC. These seven bytes are always aligned to start on a byte boundary. Third, the packet is in NRZ format.

A properly connected and configured SONIC receives every packet that is sent over the management bus, and therefore buffers the data as well as the seven bytes of status. The Packet Compression feature available on the RIC and the SONIC allows for specific handling of the data part of the packet, as described later.

Figure 1 shows the interface of one RIC to one SONIC. The SONIC is configured to run in the external decoder mode to receive the NRZ data from the management bus (refer to the SONIC datasheet for more details). The SONIC input pins CRS, RXC, and RXD tie directly to the RIC management bus output pins MCRS, MRXC and MRXD (with the RIC BINV selected for active high signals, refer to the RIC datasheet for details). The SONIC runs in Promiscuous Mode accepting all the packets from the management bus.

The packet compression output pin  $\overline{\text{PCOMP}}$  of the SONIC ties directly to the  $\overline{\text{PCOMP}}$  input pin of the RIC. The SONIC can be programmed to assert  $\overline{\text{PCOMP}}$  upon a match or a mismatch of the packet destination address with a SONIC CAM address. For managed Hub applications, the SONIC asserts  $\overline{\text{PCOMP}}$  if the destination address of the received packet does not match any address in the CAM of the SONIC. For managed bridge applications the SONIC asserts  $\overline{\text{PCOMP}}$  if the destination address of the received packet matches any address in the CAM of the SONIC.

The managed Hub application is selected in this implementation. If the packet is addressed to the SONIC, the  $\overline{\text{PCOMP}}$  pin will not be asserted by the SONIC and the RIC will not compress the data. The SONIC receives the whole packet with the seven bytes of status. If the packet is not addressed to the SONIC, the  $\overline{\text{PCOMP}}$  pin will be asserted by the SONIC. The RIC will compress the data by inhibiting the clocks during the data part of the packet, and will re-enable the clock during the seven bytes of status.

The SONIC buffers the seven bytes of management status from the RIC to memory. These bytes can then be accessed by a processor. Utilizing the packet compression technique leads to an implementation that minimizes memory requirements, i.e., buffering only the data needed by the SONIC and the seven bytes of status. The RIC contains a Packet Compress Decode Register that can be used to determine the number of bytes, post SFD, which are transferred over the management bus when the packet compression option is employed.

Since the seven bytes of status are appended after the CRC pattern, the SONIC will indicate that a CRC error occurs every time a packet is received. This should be ignored, and the SONIC should be set to save errored packets. The CRC bit in the seven bytes of status appended to the packet will indicate whether the packet has a CRC error or not.

To enable the SONIC to transmit to the network, the SONIC of Figure 1 transmits a packet to the RIC through the Inter-RIC bus. The SONIC transmit signals TXE, TXD tie to the Inter-RIC signals IRE and IRD through a TRI-STATE® buffer (74F125), which is TRI-STATE when the SONIC is not transmitting. Since the SONIC is in external decoder mode, the TXC pin is an input. An external 10 MHz oscillator provides the input to the TXC pin of the SONIC and the IRC pin of the RIC. The SONIC will drive ACTN and ACKI of the RIC as soon as it wants to transmit. Driving ACTN informs the RIC that the SONIC wants to transmit. In this implementation the SONIC is placed on top of the arbitration chain with the RIC, therefore the SONIC drives the ACKI input of the RIC when it wants to transmit.

The management bus does not experience any collisions, however any collisions on the network detected by the RIC are reported in the seven bytes of status. There will be no receive collisions on the SONIC, and the SONIC does not drive any collision signals to the RIC. The SONIC needs to be notified when a transmits collision occurs on the RIC. Therefore the COL input pin of the SONIC is driven by the ANYXNd output of the RIC whenever there is a transmit collision on the RIC and the SONIC is transmitting.

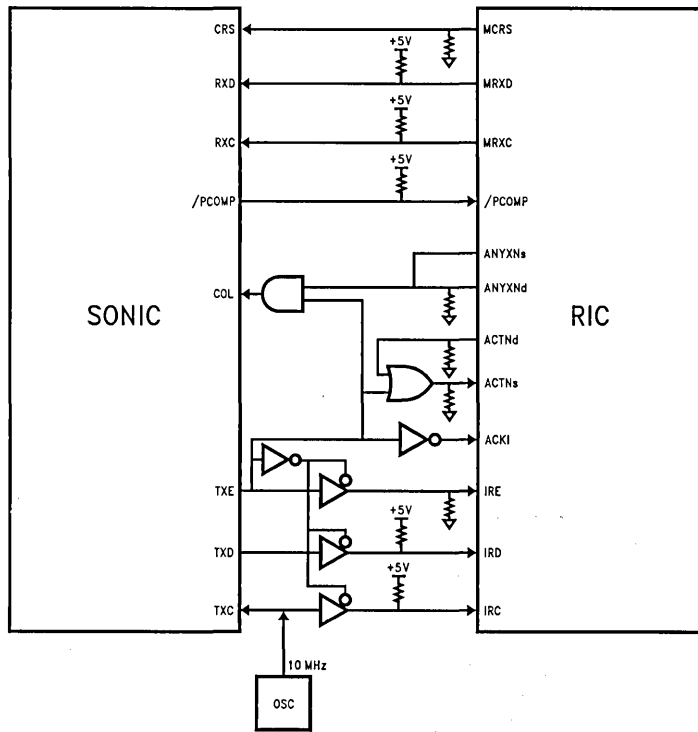


FIGURE 1

TL/F/11231-1

Figure 2 shows an implementation with several RICs sharing one SONIC for a managed Hub application. The SONIC is set in Promiscuous Mode to receive all packets, and it asserts  $\overline{PCOMP}$  upon address mismatch. The Hub is addressable, and the SONIC can receive and transmit packets to the network as well as receive the seven bytes of RIC management status. All RICs share a single management bus to send the data to the SONIC. The SONIC transmits through the Inter-RIC bus to all RICs.

To interface the SONIC to the RIC in this implementation, a PAL is needed to generate the following signals:

$$\begin{aligned}\overline{ACKO} &= \overline{ACKI} \& TXE + \overline{ACKI} \\ ACTNd &= \overline{ACKI} \& TXE \\ ANYXNd &= TXE \& \overline{ACKI} \\ COL &= TXE \& ANYXNs \\ TXEO &= TXE \& \overline{ACKI}\end{aligned}$$

This implementation utilizes the serial arbitration method, and allows the SONIC to be placed anywhere in the arbitration chain.  $\overline{ACKO}$  is asserted if the  $\overline{ACKI}$  from the RIC above it is not asserted and the SONIC wants to transmit, i.e., TXE is asserted, or if  $\overline{ACKI}$  from the RIC above it is asserted. ACTNd is asserted to tell all RICs that it wants to transmit when ACKI is not asserted and TXE is asserted. The SONIC could experience a transmit collision in this implementation since it could be in the middle of the arbitration chain. ANYXNd is asserted by the SONIC when TXE is asserted, and  $\overline{ACKI}$  is asserted by any RIC higher in the arbitration chain. The SONIC is notified of a collision if it is transmitting and any RIC asserts ANYXN. Finally, TXEO is enabled

when the SONIC wants to transmit and  $\overline{ACKI}$  from the RIC above it is not asserted.

#### RIC-NIC INTERFACE

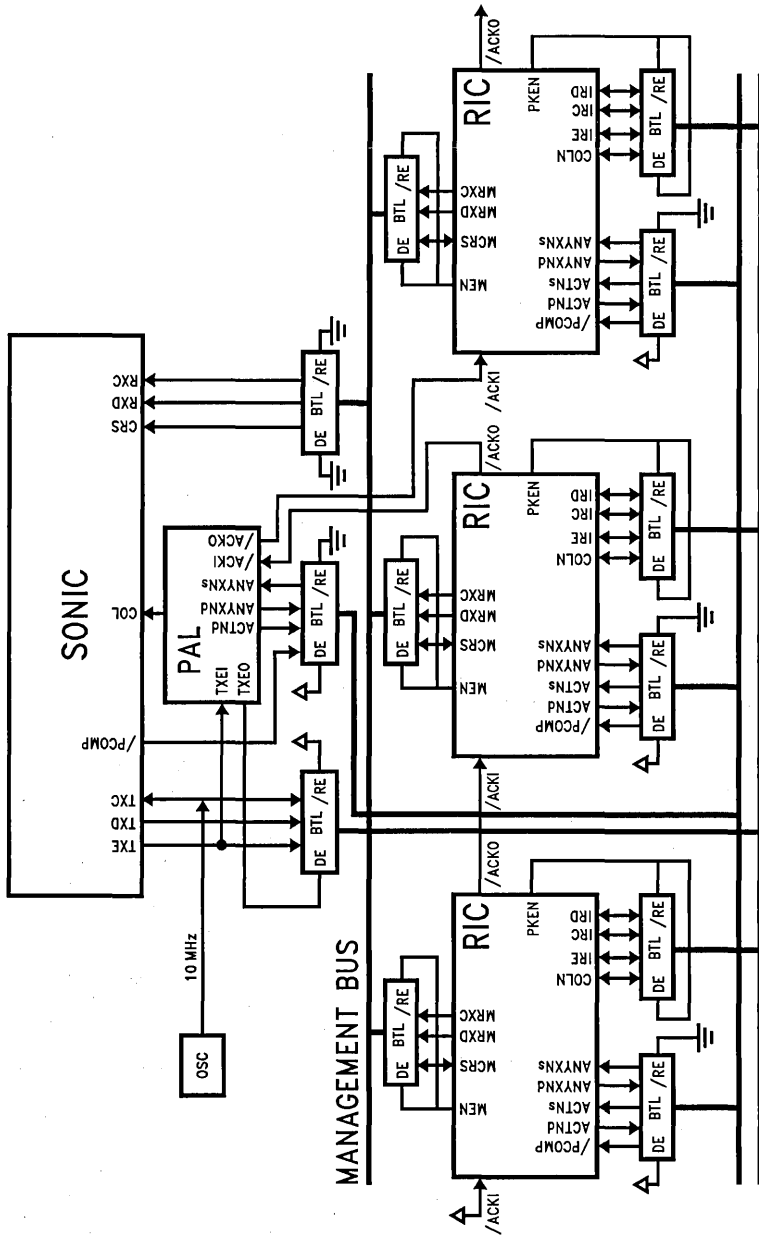
Any design that utilizes any controller other than the SONIC, such as the NIC (DP8390), to interface to the RIC should address the following points:

First, the packet compression feature of the RIC cannot be used by other controllers unless an external CAM and associated logic is used to generate the  $\overline{PCOMP}$  signal to the RIC. If this logic is available the controller may not operate properly with the clock inhibited. The SONIC has an on board CAM, and asserts  $\overline{PCOMP}$  to the RIC, and it works properly while the clocks are inhibited.

Second, due to the nature of the CSMA/CD protocol, there are situations when a collision will occur early in the packet (before SFD). This will lead to a packet transmitted onto the management bus containing only the seven bytes of status. This will be ignored by most controllers. Therefore extra logic will be required to stretch such packets to the controller's minimum acceptable packet length. The SONIC accepts such packets normally.

Third, knowing that the packet compression feature cannot be used, all packets that are transmitted over the network will need to be buffered by the controller. This requires a larger memory space, and may require a faster CPU.

Fourth, the SONIC will receive back to back packets from the management bus without missing packets due to insufficient gap (provided it is given access to memory). Other controllers may miss some packets if the gap is small.



TL/F11231-2

FIGURE 2

### OTHER INTERFACE METHODS

The method described so far to interface the SONIC to the RIC's management and Inter-RIC busses is not the only way to interface a controller to the RIC. A controller could also transmit and receive packets through the Inter-RIC bus or through any of the ports. However these two methods do not allow the controller to obtain the management bus data from the RIC. There are several drawbacks for not receiving this data:

First, even though part of the information available in the seven bytes of status is available through the CPU bus of the RIC, the CRC error flag, the Collision Bit Timer, the Repeat Byte Count, and the Inter Frame Gap Bit Timer are not available from the RIC except through the management bus.

Second, every packet transmitted through the management bus contains the number of the port receiving the packet. If the management bus is not used, the only way to obtain the port number is by setting the RIC to generate a Real Time Interrupt to the processor on every packet received. The processor then reads the Real Time Interrupt register to find out which port received this packet.

Third, in a multi-RIC system, the RIC number is essential for associating the packets with the receiving RIC and receiving port. This is included in the seven bytes of management status, and cannot be obtained otherwise directly from the RIC.

Fourth, the packet sent over the management bus contains the Source and Destination addresses, and the Packet Compress Decode Register can be used to specify the number of bytes to send over the management bus before inhibiting the clocks when  $\overline{PCOMP}$  is used. To perform this operation otherwise extra dedicated logic is needed to receive every packet on the network to read and save this data.

### SONIC EVALUATION BOARD MODIFICATION (DP839EB-ATS ONLY)

This section describes a way to interface the SONIC to receive packets from the management bus of the RIC and to use the packet compression feature, using the Repeater

Evaluation Kit (RICKIT) and a DP839EB-ATS board. A new SONIC evaluation board, the DP83932EB-AT, will not require modification. Refer to AN-855 for more information. Contact your National Semiconductor representative regarding availability.

All that is needed for the SONIC to receive the management bus data is to tie CRS, RXO, RXD and  $\overline{PCOMP}$  pins from the SONIC directly to the MCRS, MRXC, MRXD and  $\overline{PCOMP}$  pins of the RIC (see *Figure 1*). This can be achieved as follows:

1. Place the DP839EB-ATS board in external decoder mode by removing the EXT jumper in the JB2 block, and removing all the jumpers in the JB4 block.
2. Take an SNI (DP8391, or DP83910) chip and clip off pins 2, 3, and 4, and place it in the appropriate socket (U18) on the DP839EB-ATS board.
3. Solder three wires to pins 2, 3, and 4 on the back of U18 on the DP839EB-ATS board and solder the other end to a female connector attached to the prototype area of the board.
4. To utilize the packet compression feature, use a SONIC (DP83932B) (pin 26 is the  $\overline{PCOMP}$  pin). Bend pin 26 up in order for it to be accessible after inserting the SONIC back into the socket. Solder one end of a fourth wire to this pin and solder the other end to the fourth pin of the connector on the prototype area.
5. On the RICKIT (DP83950EB-AT) Main Board, solder four wires to the pin side of R31, R71, R40 and R36. Connect these wires to a female connector. These four wires should be in the proper order to correspond to the proper pins from the DP839EB-ATS board.
6. Make a four wire ribbon cable that is approximately four inches long with a male pin connector at each end. This can now be used to connect between the two male connectors on the DP839EB-ATS board and the RICKIT Main Board.

The DP839EB-ATS board now has the proper modification to receive the management bus data from the RICKIT Main Board. These boards can now be installed into the same PC-AT using the diagnostic/evaluation software provided with the board. See the software manual provided with the board, for details.

# DP83950 Twisted Pair Parametric Evaluation

National Semiconductor  
Application Note 783  
Imad Ayoub, Howard Vo  
Prasun Paul



## TWISTED PAIR PARAMETRIC EVALUATION

The following information lists the results of the Twisted Pair Parametric tests performed on the DP83950 Repeater Interface Controller (RIC™). The DP83950EB-AT Repeater Kit was used to perform the measurements. Four parts were evaluated at room temperature and 5V power supply, except where indicated.

The test results are divided into three areas; transmit, receive and miscellaneous. The tabular format used shows the parameter tested, the reference section and *Figures* of the "IEEE 802.3 10Base-T CSMA/CD Access Method and

Physical Layer Specifications" document, and the values measured on the RIC. No details for the tests/setups are provided as they follow the IEEE document specifications for each test. Additional notes and tables are included for clarification where necessary.

National Semiconductor Corporation (NSC) does not guarantee any of the values indicated in this document. The parameters indicated in the AC/DC parameters section in the RIC data sheet are the ONLY parameters that are guaranteed by NSC.

### Transmitter Specifications

| Test # | Parameter                                                                                                                                                                                                                                            | IEEE Ref. Spec. | RIC Value/Comment                                                                    |
|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|--------------------------------------------------------------------------------------|
| 1      | Peak differential output voltage: at TD± circuits terminated with a 100Ω load directly<br>±2.2V to ±2.8V                                                                                                                                             | 14.3.1.2.1      | 2.5V Peak<br>(Note 1)                                                                |
| 2      | Harmonic contents with 10 MHz signal through the transmitter<br>All harmonics should be ≥ 27 dB below the fundamental 10 MHz                                                                                                                         | 14.3.1.2.1      | Tested with a random signal, all harmonics were > 30 dB below the fundamental signal |
| 3      | Output waveform with scaling<br>Within <i>Figure 14-9</i> template                                                                                                                                                                                   | 14.3.1.2.1      | Waveforms are within template<br>Measured values are shown in Tables Ia, Ib, Ic      |
| 4      | Start of TP_IDL waveform with specified load in <i>Figure 14-11</i> and with or without cable model. The readings include idle high time and idle setting time<br>Within <i>Figure 4-10</i> template                                                 | 14.3.1.2.1      | Waveforms are within template<br>Measured values are shown in Table II               |
| 5      | Link test pulse waveform, with specified load in <i>Figure 14-11</i> and with or without cable model. Readings include amplitude and pulse width<br>Within <i>Figure 4-12</i> template                                                               | 14.3.1.2.1      | Waveforms are within template<br>Measured values are shown in Table III              |
| 6      | TD circuit differential output impedance or Return Loss spec.<br>Reflection ≥ 15 dB below incident for all power on states and for impedances of 85Ω to 111Ω                                                                                         | 14.3.1.2.2      | Within spec.<br>Measured values are shown in Table IV                                |
| 7      | TD output jitter: random signal through a 100m cable model terminated with a 100Ω load<br>Equalized for max ± 3.5 ns jitter at the end of cable model and with this equalization max ± 8 ns while TD circuit is directly terminated with a 100Ω load | 14.3.1.2.3      | Within spec.<br>Measured values are shown in Table V                                 |
| 8      | Common mode to differential mode conversion. Test circuit as in <i>Figure 14-13</i><br>≥ 29 - 17 log <sub>10</sub> (f/10) dB<br>1 < f < 20 MHz                                                                                                       | 14.3.1.2.4      | Within spec.<br>Measured values are shown in Table VI                                |
| 9      | TD circuit common mode output voltage. Test circuit is shown in <i>Figure 14-14</i><br>< 50 mV peak                                                                                                                                                  | 14.3.1.2.5      | Within spec.<br>(Note 2)                                                             |

## Transmitter Specifications (Continued)

| Test # | Parameter                                                                                                                 | IEEE Ref. Spec. | RIC Value/Comment                                                         |
|--------|---------------------------------------------------------------------------------------------------------------------------|-----------------|---------------------------------------------------------------------------|
| 10     | TD short circuit current<br>300 mA max                                                                                    | 14.3.1.2.7      | Within spec.<br>Approximately 0 mA                                        |
| 11     | TD circuit common mode impulse withstand. Test circuit as in<br><i>Figure 14-15</i><br>Impulse $E_{cm}$ applied 1000V min | 14.3.1.2.7      | Filter test—Guaranteed by filter<br>manufacturer                          |
| 12     | TD silence voltage<br>$\leq \pm 50$ mV                                                                                    | 14.2.1.1        | Within spec.<br>6 mV                                                      |
| 13     | Period of link pulses<br>16 ms $\pm$ 8 ms                                                                                 | 14.2.1.1        | 16 ms                                                                     |
| 14     | Transmit settling time                                                                                                    | 14.2.1.1        | Within spec.<br>Meets amplitude and jitter<br>specifications (2nd bit on) |
| 15     | Power cycle behavior<br>No extraneous signal on TD circuit                                                                | 14.3.2.3        | No extraneous signal on TD circuit<br>where noticed                       |

**Note 1:** The circuit used is shown in *Figure 1*. Three filters/transformer packages from three vendors were evaluated, and all of them met the amplitude required by this spec. The packages evaluated were: 1) Valor FL1012, 2) Pulse Engineering PE65431, 3) Bel Fuse 0556-3392-00

**Note 2:** The measurements were done on Valor FL1012, Valor PT3877, and Pulse Engineering PE65431. For all of these packages a 0.01  $\mu$ F capacitor is required from the center tap to ground, as shown in *Figure 2*, to reduce common mode to within 50 mV.

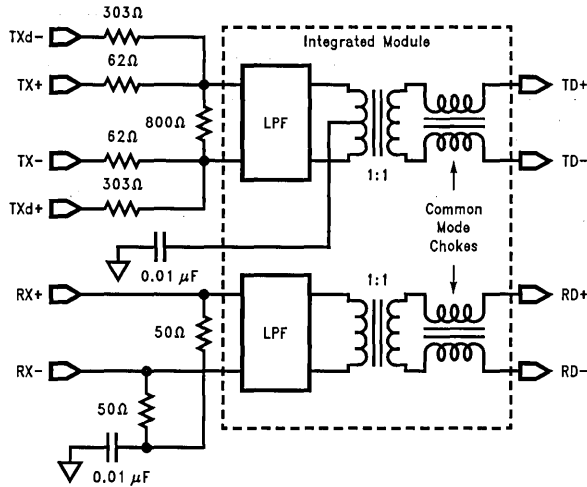


FIGURE 2

TL/F/11232-1

## Receiver Specifications

| Test # | Parameter                                                                                                                                                                                                                                                                                                                                                                                      | IEEE Ref. Spec. | RIC Value/Comment                                                                                        |
|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------|----------------------------------------------------------------------------------------------------------|
| 1      | Signals accepted by RD circuits<br><i>Figures 14-16 and 14-17 templates</i>                                                                                                                                                                                                                                                                                                                    | 14.3.1.3.1      | Test signals used did not include jitter<br>Signals accepted met <i>14-17</i> and <i>14-16</i> templates |
| 2      | Jitter accepted by receiver<br>$\geq \pm 13.5$ ns                                                                                                                                                                                                                                                                                                                                              | 14.3.1.3.1      | Guaranteed by 1 above                                                                                    |
| 3      | Jitter added by the receiver<br>$\leq \pm 1.5$ ns                                                                                                                                                                                                                                                                                                                                              | 14.3.1.3.1      | Within spec.<br>Approximatley 1.44 ns                                                                    |
| 4      | RD circuit link test pulse acceptance<br><i>Figure 14-12</i> template                                                                                                                                                                                                                                                                                                                          | 14.3.1.3.2      | Within spec.<br>Rejects <480 mV amplitude<br>Accepts down to 35 ns width                                 |
| 5      | Signals REJECTED by the receiver:<br>a) Signals that will produce 300 mV peak signal at the output of a 3 pole test filter described in A.4.2<br>b) All sinusoidal signals of amplitude less than 6.2 V <sub>p-p</sub> and frequency less than 2 MHz<br>c) All sinusoidal single cycles of amplitude 6.2 V <sub>p-p</sub> with 0° or 180° phase where the frequency is between 2 MHz to 15 MHz | 14.3.1.3.2      | Within the spec.<br>Measured values are shown in Table VII                                               |
| 6      | Idle detection by RD circuits<br>Within 2.3 BT                                                                                                                                                                                                                                                                                                                                                 | 14.3.1.3.3      | Within spec.<br>Within 2.05 BT                                                                           |
| 7      | REC circuits differential input impedance or return loss<br>Reflection $\geq 15$ dB below incident for an impedance of 85Ω to 111Ω                                                                                                                                                                                                                                                             | 14.3.1.3.4      | Within spec.<br>Measured values are shown in Table IV                                                    |
| 8      | RD short circuit fault tolerance<br>Indefinite short shall be tolerable                                                                                                                                                                                                                                                                                                                        | 14.3.1.3.6      | RD short caused no faults                                                                                |
| 9      | Receive delay                                                                                                                                                                                                                                                                                                                                                                                  | None*           | 40 ns                                                                                                    |
| 10     | Bit loss and receive delay                                                                                                                                                                                                                                                                                                                                                                     | None*           | 2.3 BT (270 ns – 40 ns)                                                                                  |

\*These are extra tests not specified in the standard.



| Miscellaneous |                                                                                                                                |                 |                                                                         |
|---------------|--------------------------------------------------------------------------------------------------------------------------------|-----------------|-------------------------------------------------------------------------|
| Test #        | Parameter                                                                                                                      | IEEE Ref. Spec. | RIC Value/Comment                                                       |
| 1             | Jabber timer                                                                                                                   | 14.2.1.6        | 5 ms                                                                    |
| 2             | Unjab time                                                                                                                     | 14.2.1.6        | Approx. 100 BT                                                          |
| 3             | Link loss timer<br>50 ms–150 ms (RIC set at 60 ms)                                                                             | 14.2.1.7        | 56 ms                                                                   |
| 4             | Polarity correction:<br>a) Inverted link pulses,<br>b) Packets with inverted TP_IDL<br>For both cases check if link pass state | None*           | Functional                                                              |
| 5             | TX output at link fail<br>No output data but link pulses                                                                       | 14.2.1.7        | Functional                                                              |
| 6             | Receiver squelch level<br>300 mV–585 mV                                                                                        | Data Sheet      | Within spec.<br>(Note 1)                                                |
| 7             | Receiver frequency acceptance: Input signal on RX $\pm$ of 1.2V to 6.2V and sweep the frequency from 0 MHz to 30 MHz or higher | None*           | Within spec.<br>Accepts >3.61 MHz and up to 20 MHz<br>(generator limit) |
| 8             | Power consumption                                                                                                              | Data Sheet      | I <sub>CC</sub> max = 350 mA (Approx.)                                  |
| 9             | Receive link_test_max timer<br>25 ms–150 ms (RIC: 32 ms)                                                                       | 14.2.1.7        | Within spec.<br>32 ms                                                   |
| 10            | Receive link_test_min timer<br>2 ms–7 ms                                                                                       | 14.2.1.7        | Within spec.<br>5.75 ms                                                 |
| 11            | Link count: lc_max<br>(RIC: 7 consecutive link counts)                                                                         | 14.2.1.7        | Functional                                                              |

**Note 1:** With a SIN wave input:

Normal mode: Guaranteed on at 520 mV, guaranteed off at 460 mV.

Low squelch mode: Guaranteed on at 360 mV, guaranteed off at 260 mV. (For use with shielded TP and extended distances.)

\* These are extra tests not specified in the standard.

TABLE Ia. Data at Different Points of the Transmit Signal at the End of the Cable Model

| RIC# | Spec. |           | Port #2    |            | Port #6    |            | Port #13   |            |
|------|-------|-----------|------------|------------|------------|------------|------------|------------|
|      | Point | Value (V) | +ve TEMPLT | -ve TEMPLT | +ve TEMPLT | -ve TEMPLT | +ve TEMPLT | -ve TEMPLT |
| 20   | A     | 0         | 0          | 0          | 0          | 0          | 0          | 0          |
|      | B     | 1.0       |            |            |            |            |            |            |
|      | C     | 0.4       | 0.75       | 0.6        | 0.58       | 0.66       | 0.75       | 0.78       |
|      | D     | 0.55      | 0.88       | 0.85       | 0.72       | 0.8        | 0.9        | 0.9        |
|      | E     | 0.45      | 0.74       | 0.83       | 0.58       | 0.86       | 0.83       | 0.78       |
|      | F     | 0         | 0.3        | 0.5        | 0.13       | 0.16       | 0.43       | 0.35       |
|      | G     | -1.0      | -0.45      | -0.37      | -0.7       | -0.62      | -0.46      | -0.54      |
|      | H     | 0.7       |            |            |            |            |            |            |
|      | I     | 0.6       |            |            |            |            |            |            |
|      | J     | 0         |            |            |            |            |            |            |
|      | K     | -0.55     | -0.96      | -0.9       | -1.024     | -0.9       | -0.97      | -0.94      |
|      | L     | -0.55     | -0.96      | -0.9       | -1.024     | -0.9       | -0.97      | -0.94      |
|      | M     | 0         |            |            |            |            |            |            |
|      | N     | 1.0       | 0.8        | 0.9        | 1.0        | 0.9        | 0.78       | 0.78       |
|      | O     | 0.4       |            |            |            |            |            |            |
|      | P     | 0.75      |            |            |            |            |            |            |
|      | Q     | 0.15      |            |            |            |            |            |            |
|      | R     | 0         |            |            |            |            |            |            |
|      | S     | -0.15     |            |            |            |            |            |            |
|      | T     | -1.0      |            |            |            |            |            |            |
|      | U     | -0.3      | 0          | -0.13      | 0.26       | -0.3       | 0.032      | -0.06      |
|      | V     | -0.7      |            |            |            |            |            |            |
|      | W     | -0.7      | -0.6       | -0.64      | 0.5        | -0.43      | -0.97      | -0.58      |

TABLE Ib. Data at Different Points of the Transmit Signal at the End of the Cable Model

| RiC# | Spec. |           | Port #2     |             | Port #6     |             |
|------|-------|-----------|-------------|-------------|-------------|-------------|
|      | Point | Value (V) | + ve TAMPLT | - ve TAMPLT | + ve TAMPLT | - ve TAMPLT |
| 22   | A     | 0         | 0           | 0           | 0           | 0           |
|      | B     | 1.0       |             |             |             |             |
|      | C     | 0.4       | 0.62        | 0.6         | 0.62        | 0.62        |
|      | D     | 0.55      | 0.74        | 0.85        | 0.62        | 0.62        |
|      | E     | 0.45      | 0.64        | 0.62        | 0.5         | 0.5         |
|      | F     | 0         | 0.26        | 0.18        | 0.04        | 0.04        |
|      | G     | -1.0      | -0.6        | -0.7        | -0.78       | -0.78       |
|      | H     | 0.7       |             |             |             |             |
|      | I     | 0.6       |             |             |             |             |
|      | J     | 0         |             |             |             |             |
|      | K     | -0.55     | -1.1        | -1.024      | -1.0        | -1.0        |
|      | L     | -0.55     | -1.1        | -1.024      | -1.0        | -1.0        |
|      | M     | 0         |             |             |             |             |
|      | N     | 1.0       | 0.8         | 0.75        | 0.9         | 0.9         |
|      | O     | 0.4       |             |             |             |             |
|      | P     | 0.75      |             |             |             |             |
|      | Q     | 0.15      |             |             |             |             |
|      | R     | 0         |             |             |             |             |
|      | S     | -0.15     |             |             |             |             |
|      | T     | -1.0      |             |             |             |             |
|      | U     | -0.3      | 0           | 0           | 0.14        | 0.14        |
|      | V     | -0.7      |             |             |             |             |
|      | W     | -0.7      | -0.62       | -0.59       | -0.38       | -0.38       |

TABLE II. Start of TP\_IDL Waveform

| Test Load                              | Amplitude (V <sub>p</sub> ) | Width (ns) | Undershoot mV | @4.5 BT (mV) |
|----------------------------------------|-----------------------------|------------|---------------|--------------|
| 155Ω // 180 μH<br>with Cable Model     | 1.28                        | 425        | -220          | -44          |
| 115Ω // 180 μH<br>without Cable Model  | 1.5                         | 431        | -500          | -36          |
| 76.8Ω // 229 μH<br>with Cable Model    | 1.05                        | 428        | -120          | -40          |
| 76.8Ω // 229 μH<br>without Cable Model | 1.27                        | 438        | -336          | -32          |

TABLE III. Measurements of Different Corners of Link Pulses

| Test Load                           | Amplitude (V) | Width at 0 to 0 Crossing (ns) | Width at 0/300 mV to 300 mV (ns) | Under-shoot (mV) | Amplitude at 4 BT (mV) | Amplitude at 42 BT (mV) |
|-------------------------------------|---------------|-------------------------------|----------------------------------|------------------|------------------------|-------------------------|
| 115Ω // 180 μH with Cable Model     | 1.6           | 333                           | 176 ns at 300 mV to 300 mV       | -80              | -48                    | -12                     |
| 115Ω // 180 μH without Cable Model  | 2.79          | 142.5                         | 140 ns at 0 mV to 300 mV         | -320             | -100                   | -20                     |
| 76.8Ω // 220 μH with Cable Model    | 1.32          | 340                           | 164 ns at 300 mV to 300 mV       | -56              | -40                    | -14                     |
| 7608Ω // 220 μH without Cable Model | 2.26          | 158                           | 152.5 ns at 0 mV to 300 mV       | -240             | -60                    | -16                     |

TABLE IV. Return Loss on the Network

| Port # | Receive      |               | Transmit (Powered Up) |               |
|--------|--------------|---------------|-----------------------|---------------|
|        | @ 5 MHz (dB) | @ 10 MHz (dB) | @ 5 MHz (dB)          | @ 10 MHz (dB) |
| 2      | -31.9        | -26.3         | -34.3                 | -23.5         |
| 3      | -39.5        | -26.3         | -32.5                 | -24.8         |
| 4      | -31.3        | -22.2         | -33.3                 | -23.0         |
| 5      | -35.4        | -24.6         | -38.9                 | -26.6         |
| 6      | -35.2        | -24.3         | -34.0                 | -22.6         |
| 7      | -30.1        | -20.5         | -36.0                 | -23.4         |
| 8      | -29.7        | -20.0         | -26.8                 | -22.5         |
| 9      | -30.7        | -20.8         | -31.5                 | -21.4         |
| 10     | -30.6        | -20.9         | -31.7                 | -21.6         |
| 11     | -32.0        | -22.7         | -36.1                 | -23.9         |
| 12     | -34.5        | -24.0         | -31.7                 | -22.0         |
| 13     | -30.5        | -21.1         | -34.0                 | -22.1         |

TABLE V. Transmit Signal Jitter at the End of a Cable Model

| Filter                    | Jitter   |
|---------------------------|----------|
| Valor FL1012              | ±1.65 ns |
| Pulse Engineering PE65431 | ±1.60 ns |
| Bel Fuse 0556-3392-00     | ±2.05 ns |

TABLE VI. Data for Transmitter Impedance Balance Test

| Frequency<br>MHz | $29-17 \log_{10}(f/10)$<br>dB | $E_{cm}$<br>V <sub>p-p</sub> | $E_{diff}$<br>V <sub>p-p</sub> | $20 \log_{10}(E_{cm}/E_{diff})$<br>dB |
|------------------|-------------------------------|------------------------------|--------------------------------|---------------------------------------|
| 1.0              | 46.0                          | 10.2                         | 28.8m                          | 50.98                                 |
| 2.0              | 40.88                         | 10.2                         | 32.0m                          | 50.0                                  |
| 3.0              | 37.88                         | 10.2                         | 35.0m                          | 49.29                                 |
| 4.0              | 35.76                         | 10.2                         | 38.4m                          | 48.48                                 |
| 5.0              | 34.18                         | 10.0                         | 41.6m                          | 47.6                                  |
| 6.0              | 32.77                         | 9.6                          | 44.0m                          | 46.7                                  |
| 7.0              | 31.63                         | 9.4                          | 46.4m                          | 46.13                                 |
| 8.0              | 30.64                         | 9.0                          | 48.0m                          | 45.46                                 |
| 9.0              | 29.72                         | 8.4                          | 48.0m                          | 44.86                                 |
| 10.0             | 29.0                          | 8.2                          | 47.2m                          | 44.79                                 |
| 11.0             | 28.29                         | 8.8                          | 84.8m                          | 40.32                                 |
| 12.0             | 27.65                         | 9.0                          | 66.0m                          | 42.69                                 |
| 13.0             | 27.06                         | 8.6                          | 38.0m                          | 47.09                                 |
| 14.0             | 26.51                         | 8.4                          | 32.8                           | 48.16                                 |
| 15.0             | 26.00                         | 8.2                          | 28.8m                          | 49.0                                  |
| 16.0             | 25.52                         | 8.0                          | 26.8m                          | 49.49                                 |
| 17.0             | 25.08                         | 7.8                          | 30.8m                          | 48.07                                 |
| 18.0             | 24.66                         | 7.6                          | 29.6m                          | 48.19                                 |
| 19.0             | 24.26                         | 7.6                          | 26.4m                          | 49.18                                 |
| 20.0             | 23.88                         | 7.4                          | 21.6m                          | 50.69                                 |

TABLE VII. Receiver Rejection Test Data

| Test #            | RIC # 20 |          | RIC # 21 |          | RIC # 22 |          |
|-------------------|----------|----------|----------|----------|----------|----------|
|                   | Port # 5 | Port # 6 | Port # 5 | Port # 6 | Port # 5 | Port # 6 |
| 5 (a)<br>@ 5 MHz  | 456 mVp  | 450 mVp  | 470 mVp  | 480 mVp  | 490 mVp  | 500 mVp  |
| 5 (a)<br>@ 10 MHz | 504 mVp  | 505 mVp  | 590 mVp  | 540 mVp  | 590 mVp  | 540 mVp  |
| 5 (b)             | 3.6 MHz  | 3.59 MHz | 3.62 MHz | 3.60 MHz | 3.60 MHz | 3.59 MHz |

# Introduction to Repeaters, the RIC™ and LERIC™ and Their Applications

National Semiconductor  
Application Note 843  
Larry Wakeman  
Bill Carlson



## INTRODUCTION

The completion of the IEEE 802.3 10BASE-T Ethernet standard has introduced the need for new products in the LAN marketplace, the twisted pair multi-port repeater. The repeater, functioning as a centralized wiring hub for the 10BASE-T star topology, is experiencing a wide variety of requirements as the number of 10BASE-T users increases. (Note: In this document the terms Hub, Concentrator, and Repeater are used interchangeably.) Some want a simple, low cost repeater that can be used in a small office environment. Other's, foreseeing a need for expansion, need a repeater that can grow with their requirements. Large companies with hundreds of nodes need a large, expandable repeater incorporating features MIS administrators can use to control a complex, enterprise wide network. With the growing need for controlling and maintaining large networks, end users are also wanting 10BASE-T repeaters that offer both basic and sophisticated management capabilities.

The LERIC and RIC repeater chips from National Semiconductor provide functions to meet a large variety of requirements for the repeater marketplace. Not only do these devices have the necessary features for implementing different management capabilities, but they also have many other important features that allow them to be effectively used in a wide variety of repeater architectures, from personal computer adapter cards to huge rack mounted systems containing hundreds of ports.

The LERIC, or LiE Repeater Interface Controller, is targeted at the smaller, cost sensitive applications, where basic management or no management at all is the only requirement. The LERIC can connect to 6 twisted pair segments and 1 thick or thin coax segment through its integrated 10BASE-T transceivers and AUI port. Statistics can be gathered from internal registers or from LEDs. It also has a simple bus for cascading many LERICs together.

The RIC on the other hand is for networks requiring full network management, from small or medium size networks expecting to expand and for the larger, corporate wide networks containing hundreds of ports. The RIC has twelve integrated twisted pair transceivers, an AUI port, and a cascading bus similar to the LERIC. It also has a management bus for easily obtaining the network statistics that are needed by high end network management software.

The purpose of this application note is to explain and define the use of 10BASE-T Ethernet repeaters incorporating the LERIC and RIC. The following subjects will be addressed in this application note:

- The role of the repeater
- Network management fundamentals
- Types of repeaters
- Basic repeater functions
- The LERIC and RIC architectures and their uses

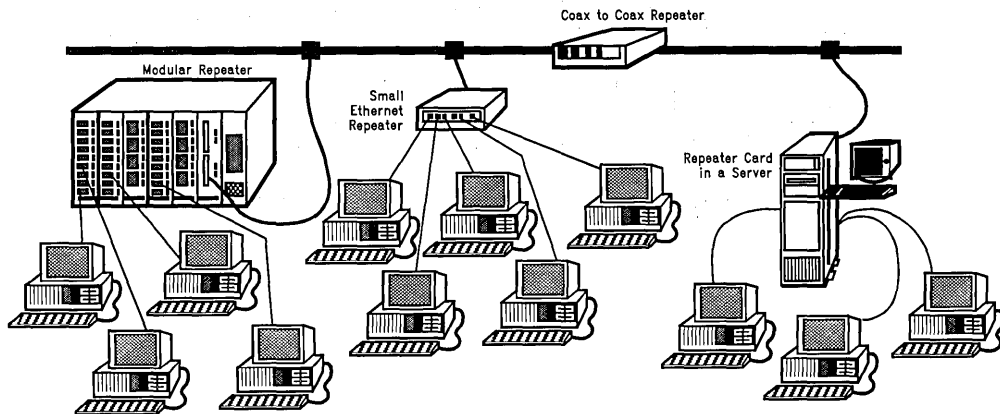


FIGURE 1. Different Types of Repeaters and Hubs

TL/F/11493-1

## THE ROLE OF THE REPEATER

The need for the multi-port repeater stems from the IEEE 802.3 architecture and standards. The 10BASE2 and 10BASE5 standards specify a coaxial cable media connected to a bus topology. 10BASE2 has a 185 meter cabling limit and 30 node limit per segment. 10BASE5 has a 500 meter length limit and 100 node maximum per segment. When the network requires longer distances or increased numbers of nodes, a repeater is necessary. While coax based Ethernet requires the repeater to be used to extend the maximum cable length, 10BASE-T twisted pair cabling requires the repeater to act as the central hub to implement its star, point-to-point topology. While it does serve to enlarge a network, its primary responsibility for 10BASE-T nodes is to allow them to access other nodes on the network. (Note: A single repeater connection is referred to as a port, i.e., a 12 port repeater can attach up to 12 cable segments.)

Figure 1 (on first page of this note) illustrates several repeaters used to expand and configure a network. There are four repeaters in this figure, each different, and each providing an example of types of repeaters most of which are described later in this paper. At the top there is a simple two port Coax Repeater; on the left is a modular (expandable) repeater for large networks; on the right is a server configured with a PC Hub Card converting a typical file server into a combined server-repeater (sometimes called a "Serpeater"); and finally in the center is a simple small repeater for a small work group. Each of these example repeaters has a port to connect to the coax cable which is used in this example as a network backbone.

Since 10BASE-T is a star topology, the repeater becomes the network center, and each port of the repeater connects to a single individual node. While the 10BASE-T network requires the repeater function, increasing the materials cost over the standard coaxial cable implementation, the above features offer many advantages over 10BASE5 and 10BASE2 cabling. These reasons for the growing popularity of this form of Ethernet are:

- Utilizes existing data grade twisted pair cabling similar wiring scheme to phone wiring.
- Ethernet can be transmitted over low cost, standard telephone wire.
- Point-to-point wiring eases cable installation.
- Distributed star has a central hub for ease of network expansion.
- Topology and media type results in low installation costs.
- The hub enables centralized network management and centralized point for connection to other communications technologies.

While MIS is interested in the financial costs of owning the network, they also want more control over their networks to maximize up-time and minimize support costs. Network management provides this. Since network management is so important, what exactly is it?

## NETWORK MANAGEMENT FUNDAMENTALS

Network Management is the process of monitoring and controlling various parameters to give administrators greater control over the networks they manage. There are 5 principle tasks and benefits of Network Management:

| Types of Management      |   |                                         |
|--------------------------|---|-----------------------------------------|
| Task                     |   | Benefit                                 |
| Fault Management         | → | Prevents network downtime               |
| Configuration Management | → | Smooths moves and changes               |
| Performance Management   | → | Makes effective use of network capacity |
| Accounting Management    | → | Tracing network utilization             |
| Security Management      | → | Protects assets and resources           |

Management gives the network administrator a wide variety of significant, practical, and cost saving capabilities. For example, with fault management, a defective or non-compliant node can be partitioned off the network to prevent consuming up valuable bandwidth, without degrading the network. Performance management helps determine when, where, and what type bridge or router to install to optimize performance on a particular segment. Charging a department for excessive network utilization could be done with accounting management.

The 10BASE-T Ethernet topology is ideal for implementing network management. Because only one twisted node is attached to a port. In this point-to-point star topology, network statistics can now be collected in the repeater because each node is mapped to a particular port. The port can be individually isolated or partitioned from the rest of the network if it is defective.

It is this simple architectural feature that has compelled IEEE 802.3 Hub Management standards to solidify. These standards enable vendors to have a common reference point and give buyers the flexibility and assurances for hardware and software interoperability. These important benefits all rely on the hub as the central component of data transmission, expandability and manageability. It should be noted that ALL network components (Bridges, Routers, Servers, and Nodes) can contain some form of network management or some mechanism to make the control, maintenance and support of each device easier. It is also ideal if all network components could "talk" the same management language to simplify the monitoring of the entire network.

Before delving into the concepts of Network Management, managing a network involves a number of activities, and network hardware can be designed to provide several levels of management for repeaters. Generally the more powerful the management functions the more costly the product to purchase, but the more automated network support (and

vendors would argue) the lower the support costs. These levels of management provided by repeaters can be broken up into three basic categories:

1. Minimal (or None): Typically no information is gathered by the repeater. There is no intelligence monitoring activities. However, generally some form of LED indicators are provided to enable visual inspection of the repeaters operation.
2. Out-Of-Band: Generally a lower cost method to gathering information than the In-Band. The Hub is intelligent and accumulates statistics. The user can only obtain these statistics through some visual alphanumeric display or more likely via a terminal attached to the hub. The major disadvantage of this technique is that it requires the net-

work manager to either physically visit the Hub to determine its operational state, or to add a modem connection to access remotely. This has reduced the popularity for this solution.

3. In-Band: This method generally can obtain the same and in most cases more information about the network than the Out-Of-Band. The major difference is that this type of repeater has a node controller that can be addressed by a remote station over the network, and information can be transferred across the network. This allows the network manager to obtain the hubs information from any network location.

Within each of the last two categories there is further differentiation by how much data is gathered as will be explained later.

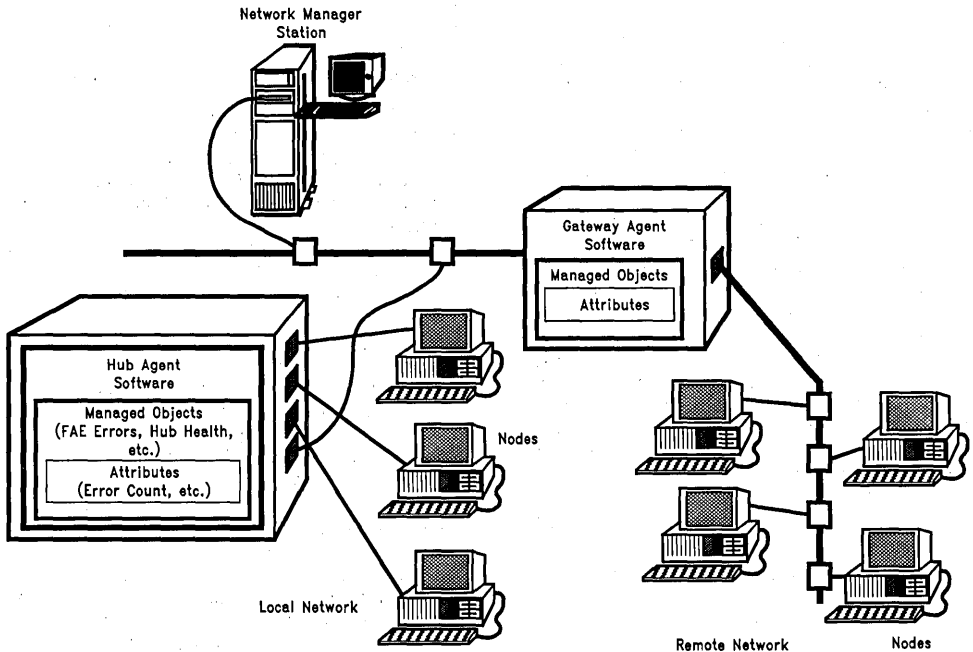


FIGURE 2. The Network Management Model

TL/F/11493-2



To further amplify the previous concepts, it is important to see how a repeater fits into standard network management mechanisms. *Figure 2* shows another typical network, this time illustrating the terms and concepts for network management. This concept applies only to a network fully capable of "In-Band" management. The terms are described below.

The *Network Management Station* is a node on the network running network management applications software. This station is where the administrator can access the managed objects. In an enterprise LAN, network management application software typically is able to control network segments other than the one it is on. For instance, in *Figure 2* the network manager software can be a node on one network segment while the managed entities are on another network segment (such as a PC on the right side of *Figure 2*).

The *Agent* is a network resource which receives commands from the manager to perform management operations and also reports status back to the manager. The agent is a separate piece of embedded software resident in the managed hub (or other device). This software communicates with the manager software via the network itself (in the case of In-Band management). Currently, standard protocols exist that enable Manager-Agent communication across multi-vendor environments. Ideally a standard protocol would allow a 10BASE-T hub agent from one manufacturer to communicate with the manager from another. SNMP (Simple Network Management Protocol) is one of these. Management application software from major manufactures use these protocols as the lower transport layer. For instance, Novell's Hub Management Interface (HMI) and Hewlett Packard's Openview use SNMP to communicate with their agents. In *Figure 2* the hub on the left and the Gateway on the right are shown to have agent software embedded in them. However, it is likely that the nodes would be running some agent software.

The entities being managed are called the *Objects*. Objects are various network statistics that are monitored and controlled by the network manager. Objects gathered by the agents depend on the type of network device (i.e., Node, Gateway, etc.). For Hubs and repeaters, objects include hub status, number of ports per hub, CRC Errors, FAE errors, number of good packets, etc. A defined set of objects are called a Management Information Base, or MIB. (Again, different pieces of network equipment can gather information for different objects, and hence support a different MIB.) For Ethernet repeaters objects are defined by the IEEE 802.3 Committee. The IEEE has standardized on the type of objects, their attributes and a database format in which an agent can present the information to a manager. As stated, this database is called a Management Information Base, or MIB. The IEEE 802.3 MIB consists of 34 attributes classified

into 3 categories called capabilities. A table detailing the specific objects, and how they are supported by the RIC and LERIC is shown at the end of this note.

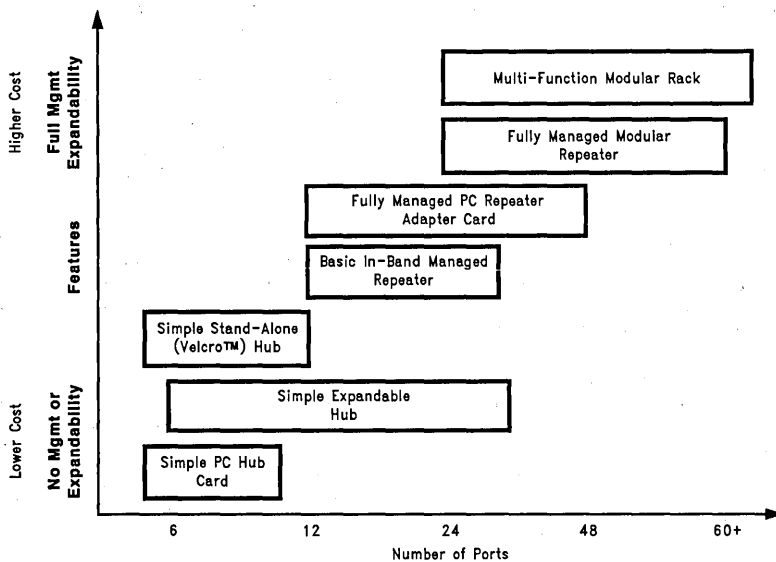
*Attributes* are parameters of an object that the agent collects on a per hub, group, or port basis. As described above for hubs, these objects include various physical layer parameters of an Ethernet node, such a CRC errors, collisions, packet length, as well as more general information such as the hub status. *Actions* are those that the agent performs on an object at the request of the manager. As an example, an action could be for the agent to partition a port from the network or to request the source address of the last packet received by the hub. *Notifications* are unsolicited reports of events that may be generated by an object. An example of a notification would be the hub agent communicating to the manager if a serious hardware error occurred.

The *Basic Control* objects consists of 19 objects which are mandatory, yet simple, for an agent to implement. Very little is required of the hardware as most of the objects are defined by the manufacturer in software. Certain key actions are partitioning off a port and notifying the manager if a port is enabled or disabled or if it has been partitioned by the autopartition state machine.

There are 2 *Address Tracking* objects that are recommended. These objects provide the network administrator with information on the node addresses and changes that occur on a port. With the hub monitoring these, it is able to map each node's address to the port it's attached to and keep track of nodes that change port location. This could be from an administrator moving cables around or from a user moving Ethernet controller boards or swapping cables. Implementing the Address Tracking category is more complex for the hub as the source address of all the incoming packets must be detected and tabulated.

While the 13 *Performance Monitoring* objects are optional, they provide the most insight into the operation and characteristics of the network. Hubs that have this capability monitor CRC errors, collisions, PLL errors as well as many more. Doing this requires a lot of hardware sophistication.

These 3 categories were chosen by the IEEE Hub Management Task Force to give hub manufacturers the flexibility to design products with 3 different price and capability levels. This was done to prevent hub manufacturers from being forced to implement all 3 categories if only a low cost Basic Control hub is required. It should be noted that if a hub agent supports one managed object in a category, then it must support them all to claim IEEE conformance of that category. For instance, if a manufacturer's hub collects the number of transmit collisions a port experiences but is not able to count the number of frame alignment errors, then the vendor can't claim to support the IEEE Performance Monitoring category.



TL/F/11493-3

FIGURE 3. Examples of Different Types of Repeaters Plotted by Number of Ports versus Features

**TYPES OF REPEATERS**

Before discussing how the RIC or LERIC is used in various repeater applications, it is very useful to look at the kinds of repeaters typically available and what they are used for.

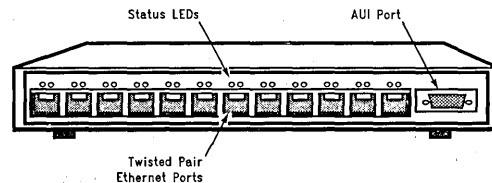
At its core a repeater function is a very simple concept (re-transmit data coming in on one port out another port). Product differentiation comes from the features added to the basic repeater function. The key differentiators are number of ports, maintainability (really network management capabilities), expandability, and ease of integration into a network. The first two features are the most important and form the axis of Figure 3. This figure breaks down the repeater types into 7 basic categories. In any given feature category other port counts than those shown are likely, however, this figure attempts to categorize the most popular configuration sizes.

As networks grow larger they require the repeater to have more features, as this diagram shows. Larger installations require a repeater to be expandable. As the network grows the repeater must grow with it to minimize duplicating equipment purchases. Having proper expansion capabilities enables this. These larger installations also require standardized network management that communicates across vendor boundaries. On the other extreme are the smaller offices where low cost and ease of use are the primary issues. These environments experience limited growth (a small dentist's office for example) so expandability is not as important either.

In the following sections we will describe the basic functions and features of these 7 repeater types.

**SIMPLE STAND-ALONE HUB**

The simple stand-alone hub as shown in Figure 4 has between 6-12 ports, doesn't have any management, and isn't easily expandable. This type of repeater is a fully self contained box, containing the repeater function, and power supply. (Note the term Velcro® hub is applied because the small size of these hubs let you stick them almost anywhere.)



TL/F/11493-4

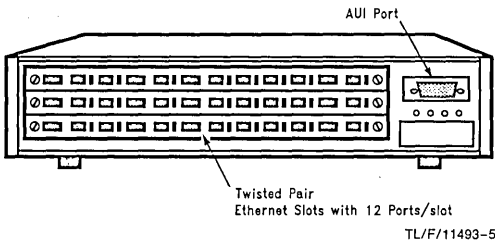
FIGURE 4. Simple Stand-Alone Repeater

The primary features of this product is its simplicity and low cost. This hub is simply intended for a small network where the users needs and understanding are simple. The user places a high value on a simple "plug-n-play" box. For maintenance and troubleshooting, this type of hub would have some status LED indicators, at least receive and collision activity LEDs for each port. This facilitates simple diagnostic on the network. These hubs typically provide an AUI (Attachment Unit Interface) port to connect to an existing 10BASE2 or 10BASE5 Ethernet LAN. The AUI can also be used to cascade other repeaters boxes if needed. However,

this method of cascading is relatively more expensive than using an expandable repeater because expansion is probably through external MAUs (Media Access Units) and the proper cabling. Expansion can also be accomplished by cascading 10BASE-T Ports but this reduces the number of available ports by 2.

### SIMPLE EXPANDABLE HUB

The simple expandable hub is essentially the Stand-Alone hub, but designed with card slots to facilitate the addition of more ports into the repeater chassis (as shown in *Figure 5*). This is used when a network is expected to grow but not too large. These hubs could support up to 24–36 nodes. The key feature of this type is its ability to expand very simply and inexpensively. These hubs typically use a proprietary bus to cascade 6 or 12 port repeaters together. This bus implementation is less costly than using coax to cascade repeaters.

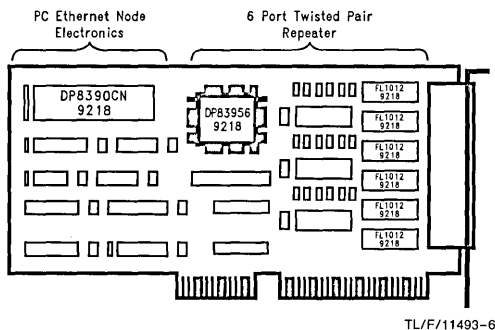


**FIGURE 5. An Example of a 36 Port Modular Repeater**

Like the Stand-Alone repeater this repeater usually implements LED status indicators, but usually does not provide sophisticated management. In some cases an add-in card for out of band management (including a CPU and RS-232 port) may be available, however most of the implementations desiring management are trending to add in the flexibility of the Managed Modular Repeater discussed later.

### SIMPLE PC HUB CARD

The concept of a PC add-in card that includes the function of a repeater is relatively new, but (due to the prevalence of PCs) provides a lot of features and benefits when compared to other options. The basic idea is to add a repeater to an Ethernet adapter, thus creating a card that when added to a PC very inexpensively turns a PC into a central control point for a typically small network. A PC equipped with this card can be used as a server-hub, or just provide a hub at less cost than the Stand-Alone Repeater (primarily because the adapter hub does not need to have a case or power supply). A typical example is shown in *Figure 6*.



**FIGURE 6. Typical Node/Hub PC Adapter**

There are two major categories of repeater cards, a high end managed card (discussed later) and a low end simple card. The low end implementation typically implements 4–6 10BASE-T ports (6 RJ45 connectors is the maximum that can fit through the back slot of a PC), and possibly include either an AUI or 10BASE2 connection. Some implementations provide a slave repeater card (containing only the repeater) that can be cascaded to the master Ethernet adapter-repeater card. This allows some expansion capability.

In addition to low cost, the advantage of this application is that user friendly utilities can be written for the PC to enable some form of management to be implemented inexpensively (usually just the Basic Control objects are supported). The major disadvantage is that if the PC is switched off then the network goes down.

### BASIC MANAGED REPEATER

One might think that a basic managed repeater without expandability would not have any application, however, there are two good applications for this repeater, and as costs for the management function drop the incremental price for added management functionality will become more popular. In one case, if the end user of a small or medium sized network is sophisticated enough he may desire a greater understanding of network health and thus need more thorough management capabilities.

Another popular application for this repeater is in a small semi-isolated work group in a sizable network. The work group itself may not require expandability, but since this work group is part of a large network then it is likely that this hub will be maintained by a central MIS organization. This organization will demand consistent hub maintenance to the rest of the network, and will require more sophisticated management than for a Stand-Alone Hub. This type of repeater looks much like *Figure 4*, but internally several component functions have been added.

Extensive management capabilities include the implementation of Performance Monitoring, Address Tracking and Basic Control MIB object tracking capabilities. Typically a CPU and a network interface controller provide the management function. This would allow the hub to be an addressable node on the network and the hub could be controlled remotely via the manager. This hub would require more memory and a larger power supply. Also the communications protocol, SNMP for example, running on this hardware platform. Typically this hub implements the IEEE 802.3 Hub Management Basic Control objects.

### FULLY MANAGED PC HUB ADAPTER

This adapter hub is conceptually similar to the low cost PC hub card, except that two major features are added: 1) Extensive Management and Diagnostics are provided, and 2) Each card supports 12 ports by using a high density connector to get the cables out of the PC, and an additional breakout box to convert to 12 RJ45 connectors.

This card's application is in high end corporate servers, and has been spurred by Novell's creation and promotion of HMI (Hub Management Interface) which provides a driver level mechanism to gather IEEE hub management objects. Due

to the large network environment a more sophisticated repeater is required as it is necessary to gather all IEEE hub management objects.

A server-repeater (Serpeater?) facilitates a number of possibilities in the corporate network environment. It enables very centralized total network services. A single box can provide not only file and print services, but can also provide routing, bridging and repeating. This potentially can ease network maintenance, and simplify configuration. However, as before the PC mechanically does not make a good repeater primarily due to the card slot form factor, and due to the limited expansion capabilities (usually it is difficult to add more than 48 ports to a server without using external repeaters).

#### FULLY MANAGED MODULAR REPEATER

For larger networks or workgroups all levels of in band network management are required. Like the Basic Managed Repeater, and the Fully Managed PC Hub Adapter, Performance Monitoring, Address Tracking and Basic Control capabilities need to be incorporated into the hub. This hub is shown in *Figure 7*.

Larger networks, 24-60 nodes, not only require full network management but now expandability is a very important issue. One can think of these repeaters as being very similar to the Simple Expandable Hub except that a high performance management agent is required and the expansion options tend to be more varied to support a large multi-vendor network.

#### MULTI-FUNCTION MODULAR COMMUNICATIONS RACK

A short conceptual jump from the Fully Managed Modular repeater is to support other communications technologies, such as Token Ring, FDDI, Routers, gateways, and poten-

tially servers. These multi-function communications equipment is for only the large networks, so extensive, computing power is required. Because of the level of network management required of these repeaters, sophisticated yet proprietary management applications software is typically offered by the manufacturer. This software would run over standard protocols however.

#### BASIC RIC AND LERIC REPEATER FUNCTIONS

The previous section focused on the feature and function differences of the various repeater architectures, highlighting their advantages and disadvantages. However, each hub contains the same basic repeater functions as defined by the IEEE. This is key as repeaters from many manufacturers need to communicate with each other and Ethernet DTEs.

This following section describes the basic repeater functions that all repeaters must have. The description is given by using the RIC/LEERIC architecture. It is possible to implement the repeater issuing different functional partitioning, however, the functions of a general repeater are basically the same. The block diagram of *Figure 8* shows the major functional blocks of a RIC or LERIC based repeater that implements the requirements of the IEEE and upon which the repeater products described earlier can be built.

It should be noticed what is *not* included in the repeater architecture. There is no complete MAC or Media Access Control unit. MACs are used by Ethernet controllers to implement the CSMA/CD protocol for gaining access to the media. Also, repeaters don't do address filtering or routing. These are done by gateways and bridges. Repeaters simply repeat the data that is received from one port and transmit it to all the others. The repeater has to re-time the received packets and remove accumulated jitter. The repeater must also not allow defective nodes to consume network bandwidth.

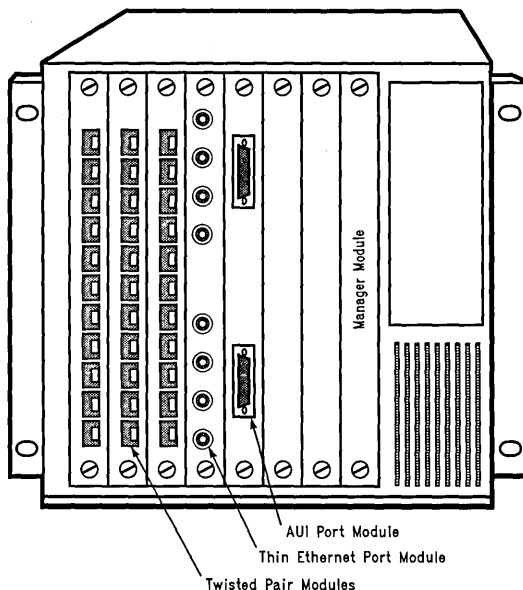
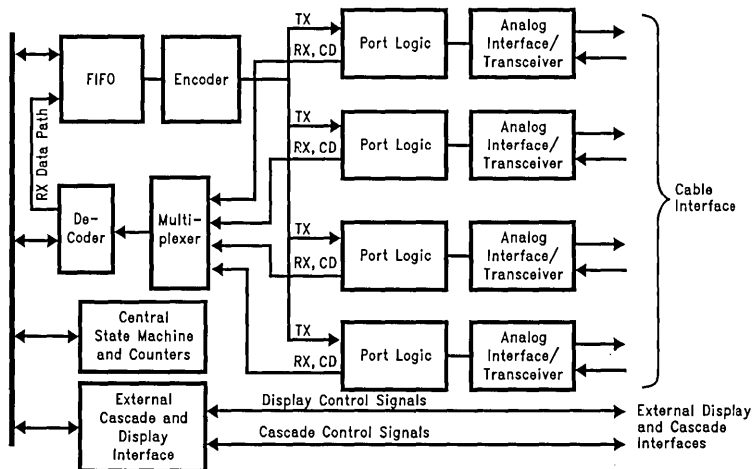


FIGURE 7. Typical Modular Repeater Including Options for Multiple Cable Media and Network Management

TL/F/11493-7



TL/F/11493-8

FIGURE 8. Simplified General Block Diagram of the Core Repeater Functions of the RIC and LERIC

### PORT SPECIFIC FUNCTIONS

Within a repeater there are functions that are shared by all the ports and those that each port duplicates for itself. As the diagram shows, there are 5 functional blocks that are identical for each port.

**Analog Interface/Transceivers.** This block is not actually an 802.3 requirement. In a general repeater this function is required to connect the repeater to the media, and is generally incorporated into the repeater box, but it is not actually part of the repeater standard. Most integrated repeaters implement either a transceiver interface or all or part of a transceiver. In the case of the RIC/LERIC, this block includes all functional and electrical specifications to interface to a particular transmission media (coax, twisted pair, fiber). It should be noted that with the exception of the transceiver, everything about the repeater is independent of what media is attached to the ports. When the transceivers are for twisted pair, they monitor the link integrity of the attached segment.

**Port Logic.** This block performs many different functions. There are two different state machines for each port. One is called the Port State Machine, or PSM. This state machine is linked with all the other PSMs to perform port arbitration. This arbitration is needed to determine which port should be the source of data or collision information when multiple ports receive data simultaneously. The winning port is called Port<sub>N</sub> or Port<sub>M</sub> depending on the type of activity. Port<sub>N</sub> is defined as the highest priority port experiencing receive or collision activity. Port<sub>M</sub> is defined as the highest priority port that is last experiencing a collision. This state machine is also used to detect collisions and indicate them to the other ports. The second state machine imple-

ments the port auto-partitioning algorithm. When a port experiences more than 30 consecutive collisions, this state machine prevents any further data on this port from being repeated on the network. This effectively blocks it off. The port is re-enabled when a packet is successfully received or if a packet is transmitted to it.

The port logic also monitors a port to determine if a transmission exceeds a specified limit, and if so turns off that port so it doesn't bog the network down. The port is re-enabled after a specified time period has elapsed since the transmission ended.

### CENTRAL REPEATER FUNCTIONS

All repeaters tend to have some functions common to the individual ports implemented as a central function. The RIC/LERIC are typical implementations and implement the following blocks as a central function.

**Multiplexer.** This function multiplexes the packet from Port<sub>N</sub> to internal functions in the repeater.

**Decoder/FIFO/Encoder.** The Decoder/FIFO/Encoder plays an important role in the recovering and re-timing the Ethernet data. As a signal travels from the DTE to the repeater several factors degrade the quality of the signal, and the repeater must remove these distortions before re-transmitting the data.

1. The signal transmitted down the Ethernet media accumulates jitter due to the different impedances, noise, and discontinuities in the cable.
2. The preamble of the receiving packet is shortened. This caused by the signal being attenuated and delays in the squelch circuitry being activated.

This block helps to eliminate these degradation. It has a phase lock loop which removes jitter. The packet is then sent to the FIFO. The FIFO buffers the data portion of a packet until a proper length preamble can be transmitted by the ports. The FIFO also compensates for data rate differences between the node and repeater. The encoder puts the packet back into manchester form, but now encoded without jitter and at the proper frequency.

**Central State Machine and Counters.** These functions form the heart of the repeater. They control the port logic and the majority of data and collision propagation operations as defined by the IEEE specifications. This block insures minimum packet fragment length and controls the FIFO to insure that the preamble is of the specified length. Collisions are also handled here. When a port is repeating a packet and senses activity on its RX ± pair a transmit collision will result. The central state machine will transmit a jam pattern to all the ports to inform the attached nodes of a collision. This block also implements other IEEE defined timings and functions central to the repeater function.

**Display Devices and Drivers.** While not required by IEEE, some form of status display information can provide indication of repeater health. This block takes various signals from within the repeaters and makes them available to the display.

**Cascade Logic.** For a modular repeater this logic provides the key internal arbitration and data signals externally to facilitate the addition of additional repeaters/ports to the repeater system.

### THE LERIC AND RIC ARCHITECTURES AND THEIR APPLICATIONS

The LERIC and RIC are National Semiconductor Corporation's solution to the implementation of IEEE 802.3 compatible multi-port repeaters. The LERIC and RIC offer all the features needed to address this marketplace. One of the most important features is the ability to support network management. The LERIC and RIC offer different levels of network management. As stated at the beginning, the LERIC is focused toward applications in small networks where basic management or no management at all is required. The RIC, on the other hand, is ideally suited for large networks where performance monitoring in addition to basic management capabilities are required.

### THE LITE REPEATER INTERFACE CONTROLLER

The LERIC is a fully IEEE compliant repeater using as its core the general repeater architecture described previously. It adds all the features needed to be effectively used in its intended applications: the smaller networks where limited management is the only requirement. In these applications, Basic Control management is easily accomplished with the LERIC.

The LERIC, in addition to the basic repeater blocks discussed earlier has a number of specific feature blocks that are described in the following sections.

**Six Twisted Pair Transceivers and AUI Port.** 10BASE-T transceivers are integrated onto the LERIC to save board real estate. The transceivers meet the IEEE 802.3 10BASE-T specifications. The transceivers can be disabled, and turned into pseudo-AUI ports for connection to coax or fiber transceivers. While not fully AUI driver level compatible, they can drive a short distance on a PCB for connection to these alternate transceivers. The LERIC also has a single fully IEEE compatible AUI port that can drive a standard 50 meter AUI cable or can connect to a coax or fiber transceiver directly on the PCB. This port is typically used to enable the LERIC to connect to a network backbone.

**CPU Bus.** The LERIC has a bus so a CPU can access internal registers of the LERIC. One of these is a status register that indicates if any port is experiencing a reception, collision, partition, or if the LERIC is jabbering. In addition to the LERIC status register which indicates status for the entire repeater, each individual port has its own status register, called the Port Status and Configuration Register. Each port can be configured through this register. A port can be disabled and the squelch level be reduced to handle special cable conditions.

The CPU bus is multiplexed to provide the signals to perform the Mode Load self-configuration and is the pathway for the LED status signals as described below.

**LED Display.** The LERIC provides information to driver LEDs through the multiplexed operation of the CPU bus. Low cost 74LS259 addressable latches are used externally to latch the CPU bus and drive the signals to the LEDs.

The LEDs are used for visual monitoring of the repeaters status. There are two modes of LED display in the LERIC. In maximum mode, 5 LEDs are provided for each of the twist-

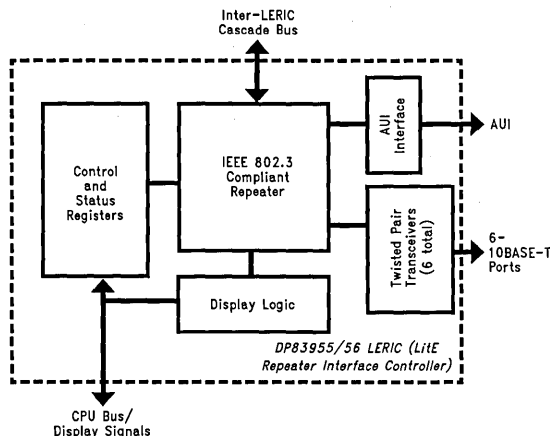


FIGURE 9. Simplified Block Diagram for Lite Repeater Interface Controller

TL/F/11493-9

ed pair ports to indicate reception, collision, partition, polarity, and link status. The AUI port has LEDs for collision, reception, and partition status. There is also an LED to indicate jabber status of the repeater.

**Mode Load.** An important feature of the LERIC is its ability to perform a hardware self-configuration. Simple pull-ups and pull-downs are attached to the CPU bus in a configuration suited for the particular application. When the mode load signal is asserted, the bit pattern defined by these registers is loaded into the LERIC and it is configured. This operation is typically done for power-on configuration of the LERIC, but can also be used whenever LERIC needs to be reset and reconfigured. Options such as twisted pair or pseudo-AUI port definition, external PLL, LED display mode, and others are loaded here.

**Inter-LERIC Bus.** The Inter-LERIC bus is used for cascading multiple LERICs and interfacing to a network controller for In-Band hub applications. There is often the need to have more than 6 ports and/or a network controller so having a simple, but powerful way to expand is very important. The Inter-LERIC bus consists of three groups of signals:

- **Port Arbitration Signals.** These signals provide a way for multiple LERICs to arbitrate for Port\_N and Port\_M status. The port arbitration signals essentially connect the arbitration logic of the port state machines together.
- **Status Signals.** The status signals indicate receive data activity and collision status of the network and communicate it to the different LERICs.

- **Data Signals.** These signals actually transfer the repeated packet from the LERIC containing Port\_N to all the other LERICs and to a network controller if there is one in the system. Packet data from the receiving LERIC is decoded from manchester and put into serial NRZ form before being driven onto this bus. The rest of the LERICs read the data and encode it back into manchester before transmitting the data to the ports.

There are two versions of the LERIC, and the major difference is the implementation of the Inter-LERIC bus. On the DP83955, the bus is designed with fewer signals intended for limited cascading primarily on a signal card. The DP83956 has the same bus as the RIC (DP83950) and therefore can be easily externally buffered and cascaded over large buses or many cards.

**LERIC APPLICATIONS**

The LERIC fits in designs at the lower end of the Port and Feature spectrum of *Figure 3*. These applications are described in the following.

**Simple Stand-Alone Hub**

This design is very straightforward. In the example of *Figure 10*, a 12+2 Hub, two LERIC's are used. These two chips are cascaded directly through their Inter-LERIC bus. The media interface to twisted pair is very simple requiring only a few buffers, filters and transformers. The status indication for each port is displayed via an LED array, this array is connected to each LERIC's CPU/LED bus, and feeds some 74LS259's which drive the LEDs.

This simple interconnection of 2 LERICs and minimal external logic enables the design of a very compact hub.

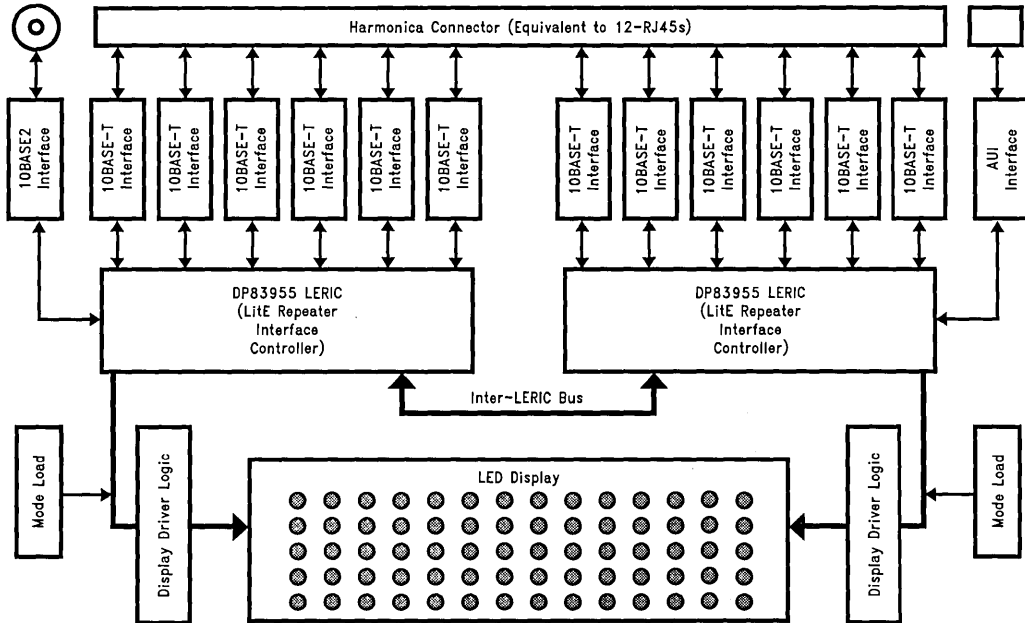


FIGURE 10. Simple 12 Port Twisted Pair Hub with both a Thin Cable and AUI Connection

TL/F/11493-10

**Simple Expandable Hub**

This application builds on top of the previous simple hub, and adding some buffer logic onto the Inter-LERIC bus. The biggest design change is really mechanical. Rather than a single PCB design as in the Stand-Alone Hub, this design is usually based on some form of card cage or plug-in slots.

Figure 11 shows a block diagram for a 6 port module that could be plugged into the card cage with other similar modules to make this a versatile setup.

**Simple PC Hub Card**

This repeater takes advantage of the PCs power supply, enclosure and CPU. As mentioned this architecture takes advantage of the PC to provide a very flexible repeater and adapter card combination. The DP8390 Network Interface Controller with its buffer RAM and bus interface provides the MAC to the network and buffers packets to a local memory for later processing. This allows the card-PC to act as a typical network attached computer. This implementation

also connects the LERIC's Registers to the PC bus, enabling PC based software to provide basic managed objects.

Typically this type of design has between 4-6 RJ-45 ports. Six is the maximum number of RJ-45s that can be accessed through the standard PC's back slot opening. The AUI port interface shown in Figure 12, is typically brought out a separate slot if needed.

The cascade port is usually connected to other cards via a ribbon cable. When more ports are required, additional slave LERIC adapter cards can be cascaded to the main master card through the buffered Inter-LERIC bus (top of Figure 11).

Unlike previous examples, the display interface is typically very simple. Minimum mode display give some general purpose display which is typically used for installation diagnostics. (For run time diagnostics a software implementation can be developed to display the "LED-like" symbols on the PC's display.)

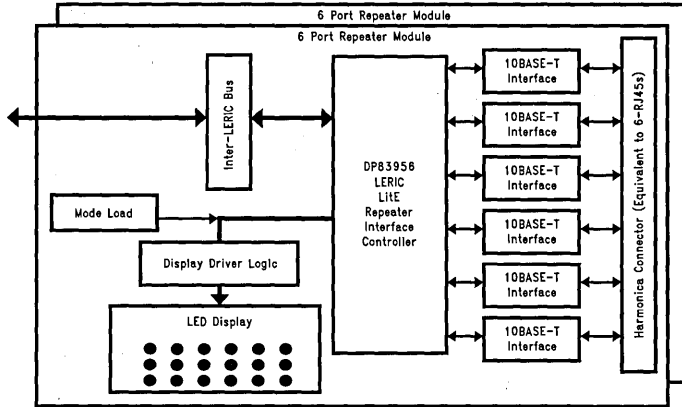


FIGURE 11. Module Hub with 6 Port 10BASE-T Modules

TL/F/11493-11

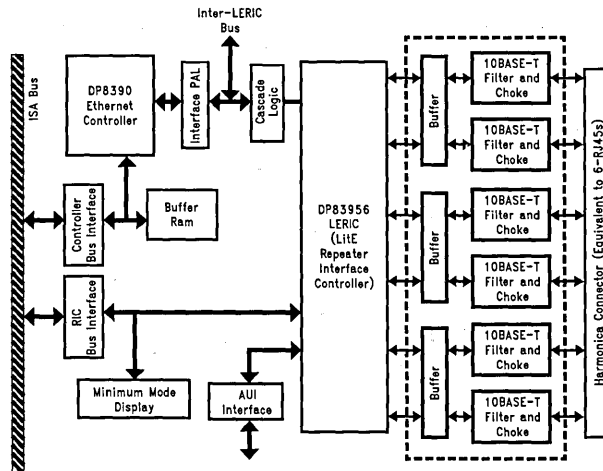


FIGURE 12. Simple PC Hub Card Block Diagram

TL/F/11493-12



### Low End In-Band Managed Hub

There are many applications of the LERIC where it is focused at smaller networks under 24 ports where full network management isn't needed. This repeater configuration supports the Basic Control capability of an IEEE MIB. The LERIC is able to turn ports on and off and indicate their status to a requesting manager. Address Tracking is possible, by using the NIC in promiscuous mode. Performance Monitoring is not possible with the LERIC because it doesn't have the capability to gather all the required physical layer statistics.

As can be seen from *Figure 13* the block diagram for this type of hub is very similar to the PC Hub card except that a CPU and RAM/ROM are added, and the port restrictions of the PC do not apply. The internal registers of the LERIC are easily accessible by the host CPU so management information can be easily collected.

### THE REPEATER INTERFACE CONTROLLER

The DP83950 RIC is a high end, feature rich device which implements all the required functions of an IEEE compatible repeater along with many additional features that enable it to gather all the mandatory, recommended, and optional IEEE capabilities for network management. The RIC is intended for networks requiring full network management now or the need for it in the future.

To compare the RIC and LERIC, they both have a cascading bus for expandability. In fact, the Inter-RIC™ bus and the Inter-LERIC™ bus are identical between the DP83950 and the DP83956, and can be connected together (The DP83955 Inter-LERIC bus is slightly different but still compatible).

Another difference between the RIC and LERIC is that the RIC contains 12 transceivers verses the LERICs 6. However, like the LERIC these 12 RIC ports can be selected to be configured as either pseudo-AUI port or twisted pair. The RICs internal twisted pair transceivers are identical to the LERICs.

Both the RIC and LERIC have the same modes of LED display. Like the LERIC, the RIC also can be configured with the Mode Load operation, but it is more likely to be configured by a CPU that typically resides in larger hub configurations.

By far the most outstanding difference between the RIC and the LERIC is their level of network management statistics gathering capability. Because the RIC is focused at the larger networks where full management is required, it has a much wider array of management components. There are two sources of network statistics in the RIC:

- Status, Event Record, and Event Counting registers, and Interrupts
- Management bus

**Internal Registers.** Like the LERIC, the RIC has status registers that give repeater status and individual port status. In fact, these registers are very similar except the RIC provides more port status information. What makes the RIC different are the Event Record and Event Counting Registers, and the two interrupt pins. Each port has one each of these registers and they collect the important physical layer statistics that are needed by the IEEE Performance Monitoring objects.

Events in the status, Event Record and Event Counting can generate interrupts to the CPU through either the Real Time Interrupt (RTI) or the main Interrupt pin. This facilitates real time statistics gathering.

The Event Counting register counts a single network event on its every occurrence. One of 11 events are available to chose from which is done by software through a mask register. This function is particularly useful to count a rapidly occurring event, such as collisions. When the counters reach one of several chosen thresholds, they can interrupt the CPU.

The Event Record register is more flexible but requires more attention by the CPU as it provides real time status information. This register can be configured to log the occurrence of up to 8 events in a byte wide register. Not all 8 need to be logged and can be chosen through a mask register. Every time an unmasked event occurs an interrupt can be generated.

The other registers in the RIC allow software to quickly isolate which port is the source of activity without having to poll each register.

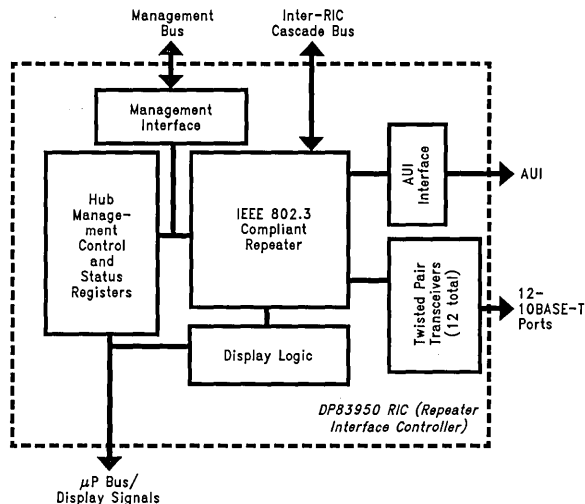


FIGURE 14. Simplified DP83950 Block Diagram

TL/F/11493-14

**The Management Bus.** Collection of a wide variety of network statistics is a major feature of the RIC, and the management bus is the RIC's most powerful pathway to communicate this information to an intelligent repeater system. The RIC couples to an Ethernet controller to form an intelligent, In-Band repeater. The management bus is similar to the data signals of the Inter-RIC bus. What makes the management data signals different from the Inter-RIC bus is that per packet statistics associated with the currently repeated packet are appended to the end of the packet on the management data signals, as shown in *Figure 15*.

The information sent on the management bus is contained in the seven additional bytes the RIC appends to the end of the packet, as shown in *Figure 16*.

These seven status bytes are on a serial data bus and are sent to main memory by adding a custom circuit, or more typically by a network interface controller. This can be accomplished using the DP8390 NIC or the SONICTM. There are advantages and disadvantages to all three approaches.

Designing a custom de-serializer circuit which converts the serial management data so that can be read by the host CPU takes design time but the interface can be tailored to the systems interface of the repeaters architecture. Since the repeater is In-Band, and Ethernet controller is still needed somewhere in the system.

Secondly, the NIC offers a low cost solution to buffering the management information to memory. A somewhat faster CPU (than in the first or third options) may be necessary to ensure that the NICs buffer does not overflow due to the large number of packets received by the hub.

The third solution is to connect the higher performance SONIC controller to the management bus. The SONIC pro-

vides signaling to enable the use of a special feature of the RIC-SONIC interface which is the ability to optimize packet storage and bus bandwidth by eliminating the unnecessary data field from packets (except for management packets which are addressed to the SONIC where the data field must be retained). This feature is called Packet Compression.

There are several advantages to the management bus:

- In a multi-RIC repeater, the management status bytes are mostly available from one source. This saves a processor from having to read data from a multitude of sources.
- CPU performance requirements may be reduced since using the management bus for gathering of network statistics and buffering from the CPU eliminates most of the real time processing required when statistics are gathered entirely by using the RIC's registers.
- The management bus records interframe gap time which allows the administrator to see if there are any nodes that violate this important IEEE spec.
- When using a SONIC controller, packet compression can be employed, and this can further reduce system overhead, by eliminating the buffering of packet data. The SONIC only has to buffer the 21 bytes (7 status + dest. address + source address + type/length field) in 6 32-bit write operations. This can be done very quickly, (<1  $\mu$ s).

The management bus architecture is particularly cost effective for in band management hubs. These hubs will require an Ethernet controller to communicate to the Network Manager, and in this case the use of the management requires no addition logic to implement.

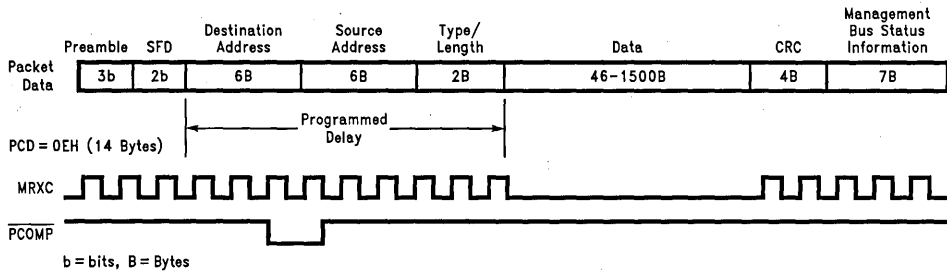
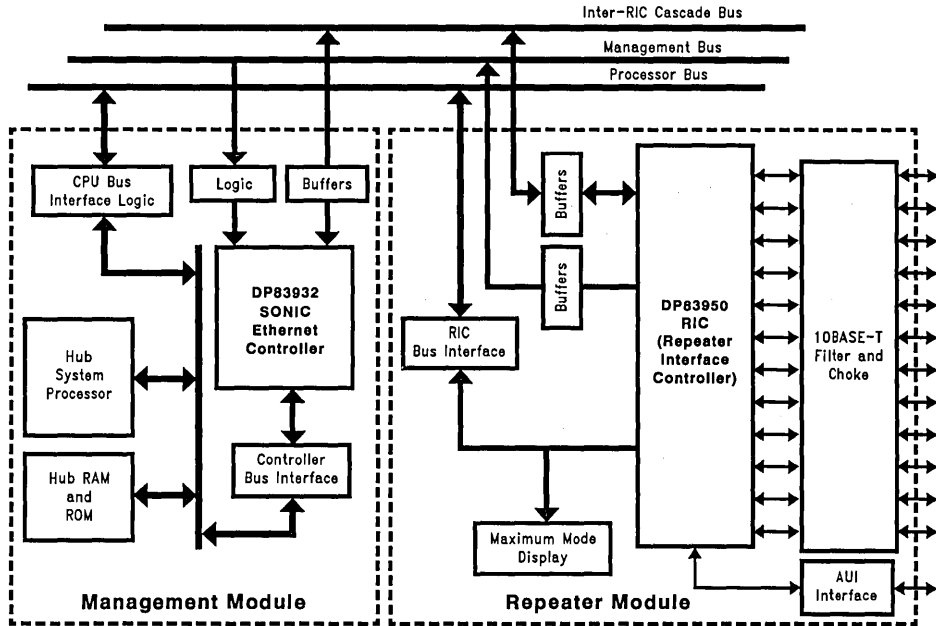


FIGURE 15. Signals Appearing on the Management Bus

TL/F/11493-15

| Summary of Management Bus Statistics                                                                                           |
|--------------------------------------------------------------------------------------------------------------------------------|
| RIC No. and Port No. Packet Received On                                                                                        |
| CRC Error, Frame Alignment Error, Tx Collision, Collision, Short Event, Late Collision, Non-SFD, PLL Error, FIFO Error, Jabber |
| Collision Bit Timer                                                                                                            |
| Packet Data Byte Count                                                                                                         |
| Repeater Byte Count                                                                                                            |
| Inter-Frame Gap Bit Timer                                                                                                      |

FIGURE 16. Summary of Management Bus Information



TL/F/11493-16

FIGURE 17. Block Diagram for Managed Repeater Showing Multiple Repeater Cards and a Management Module

### RIC APPLICATIONS

Like the LERIC, there are many applications for the RIC. The primary (though not only) applications for the RIC are in high end hub applications where full support of hub management IEEE objects are required. The large rack mounted system is an ideal application for the RIC for high end repeaters.

### FULLY MANAGED REPEATER

For networks not quite so large but needing management, a fully enclosed module running SNMP with 12-24 ports implements a low cost hub yet provides all the management capabilities of larger systems. This is called the Fully Managed Hub. This system block diagram is very much the same as for the Modular Managed Hub, except that a single non-expandable 12-24 port PCB contains the repeater function, and Ethernet Controller, CPU and memory. Functionally, this system's block diagram is very similar to the Managed Modular Hub of *Figure 17*. Hence the functional description is the same as the modular managed repeater described next.

### MANAGED MODULAR REPEATER

The modular managed repeater example is a rack mounted repeater, containing independent repeater modules. Since these repeaters could support up to hundreds of ports, it must be easily expandable. The modules are typically stacked vertically or horizontally in a chassis. Sophisticated In-Band network management is required and so an Ethernet controller with a CPU usually on a separate module is required. The ideal controller is the SONIC directly connected to the RIC. Expandability is very important so the hub can grow along with the network.

In *Figure 17*, the block diagram of the management module is shown on the left, and one of several repeater modules block diagram is shown on the right side. The backplane for this modular repeater is actually the center of the hub. This backplane usually consists of 3 buses. First, a CPU bus which allows the management module's CPU to control the repeater modules. Second, a management bus which is an extension of the RIC's management bus, and allows the management module's SONIC to access the RIC statistics information. Third, the repeater also has a repeater cascade backplane for connecting multiple repeater modules into a single logical repeater.

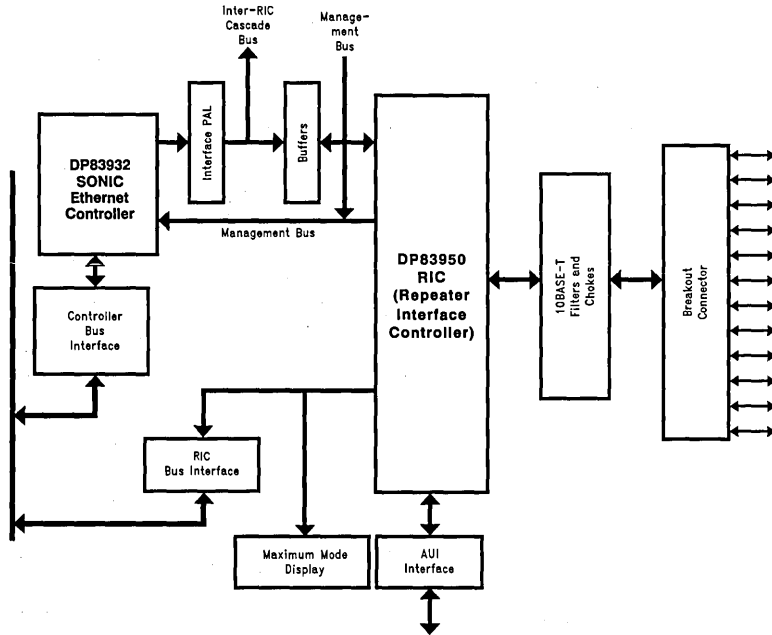


FIGURE 18. Block Diagram for Fully Managed PC Hub Card

TL/F/11493-17

**FULLY MANAGED PC ADAPTER**

This repeater application has become popular since the development of Novell's Hub Management Interface (HMI) specification. This driver specification provides a standard software method to obtain all the network management capabilities of the larger, self-contained hubs, but hardware is much simpler and less costly (if you assume that the cost of the PC is not included). The block diagram of this adapter card is very similar to the LERIC/NIC solution except that a RIC is typically used in conjunction with a higher performance Ethernet Controller such as a 16-bit or 32-bit SONIC, as shown in *Figure 18*.

Since many servers are based on the EISA or Micro Channel bus, a 32-bit SONIC could act as a high performance bus master. An ASIC and/or other logic provides the interface between it and the system. On the network side, the SONIC's PLL/ENDEC is disabled and the management bus connected to the receive signals and the Inter-RIC bus connected to the transmit signals using a simple PLD incorporates some of the needed glue logic.

12 twisted pair ports of RJ45's are physically too large to fit into one PC expansion slot opening. So usually the port connection is made via a high density 50 pin (like a SCSI II) connector, and a cable connects to a breakout box, containing a 12 position RJ45 connector and typically the LED array.

The breakout boxes are usually placed on the floor or in some sort of standard rack, *Figure 19*. Typical configurations support up to 48 10BASE-T nodes with an AUI port for attachment to a 10BASE5 or 10BASE2 network. Expansion limitations are primarily due to the limited PC slot configurations.

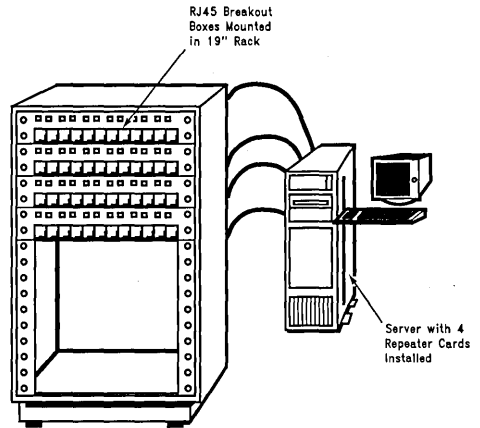
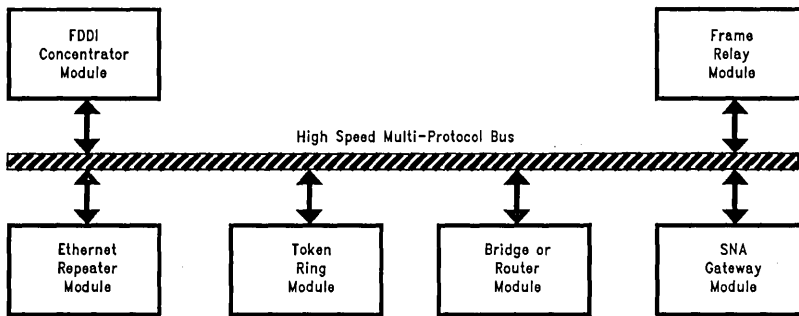


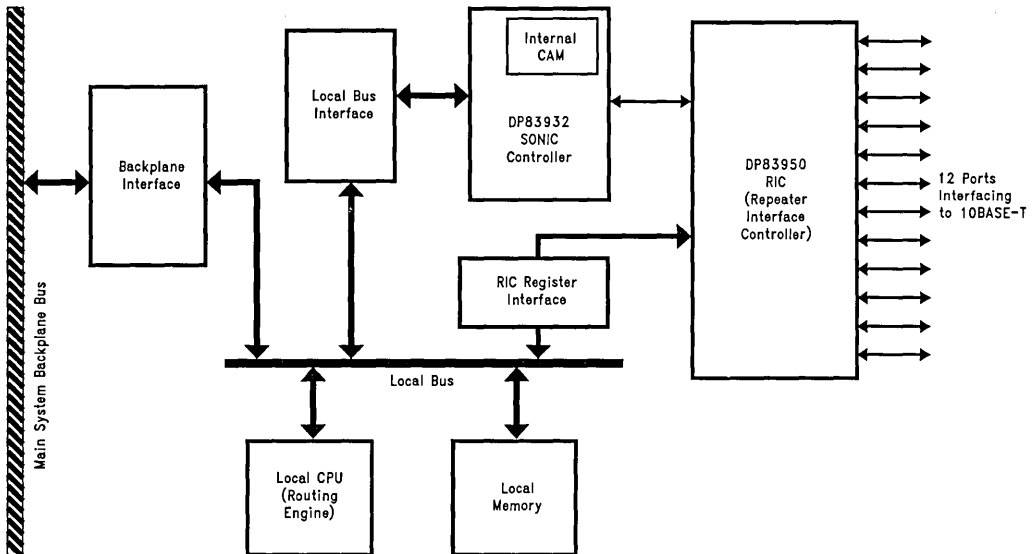
FIGURE 19. Breakout Boxes out of the File Server

TL/F/11493-18



TL/F/11493-19

FIGURE 20. Possible Modules for Communications Rack



TL/F/11493-20

FIGURE 21. Bridge/Router Combined with a Repeater (Using Hardware Address Filtering)

Vendor's offer a lot of flexibility in server configurations by offering slave adapters. These adapters are repeater boards without the controller. This allows the creation of single large repeaters with a single controller, or to use multiple controllers creating several networks connected by the server's bridging software.

Using Novell's Remote Network Interface, In-Band managed hubs can be located in other nodes besides the server. This allows repeaters to be distributed around the network where workgroups are more concentrated. Management can be done remotely from any node. Using this broad systems approach allows very large managed networks to be constructed at reasonable costs.

For expansion, additional RIC slave cards can be cascaded to the master card through the Inter-RIC bus and management bus. All of the RICs in this system are configured by software.

#### MODULAR COMMUNICATIONS RACK

For the purposes of this document, the Communications Rack is really a modular multi-function box (typically placed on a rack in a wiring closet) that in the ideal case, can support and LAN (or Wide Area Network) connection function. As shown in *Figure 20* this box could support not only repeater and management functions but bridge router, WAN connections, SNA Gateways, or possibly even file and printer server functions.

When using the RIC in this type of equipment, the architectures are very similar to previous modular repeaters, except that the bus supports more sophisticated functions, probably the rack's backplane is actually a high speed 16-bit-32-bit parallel data/control bus that routes network data between modules in a very sophisticated manner. The details of the architectural for such a box is beyond this paper, as it would involve discussions beyond applying the RIC or LERIC. However, there are a couple of interesting applications for the RIC within this box.

The most interesting one is the bridge/router-hub module, as shown in *Figure 21*. In this application, which does use the RIC-SONIC combination yet again, utilizes the SONIC'S internal CAM as an address filter. In this application a single RIC connects point-to-point to the RIC. Since only 12 nodes would typically be connected to the RIC, the SONIC'S internal CAM can perform the address filtering. If a situation were such that the RIC would be connected to more than the 12 nodes, then either software or an external CAM would be necessary to do address filtering.

#### **RIC AND LERIC REPEATERS FOR ANY APPLICATION**

In this note, a wide variety of applications have been introduced in a general systems oriented overview.

A sampling of the breadth of possible hub/repeater applications has been presented as well as specifically highlighting the importance of network and Hub management as a required feature of many repeaters. Discussions of the operational characteristics of National's repeater family have shown how these repeaters have features that can be effectively utilized to build systems that address any possible Ethernet network repeater product.

The RIC provides a very feature rich IC platform that enables building fully managed very high functionality repeaters of various styles and architectures.

The LERIC is a very cost effective simple repeater IC that should appeal to simple non-managed repeater applications that are typically provided for small cost sensitive LAN applications.

With both these devices low end simple Velcro hubs, PC Hub Cards, Managed Hubs, Modular Repeaters all can be designed very easily and cost efficiently by selecting and using either the RIC or LERIC.

**APPENDIX A.  
IEEE 802.3 HUB MANAGEMENT IMPLEMENTATION**
**Management Criteria**

A: Basic control capability      Mandatory  
 B: Performance monitor          Optional  
 C: Address tracking capability    Optional

**HUB MANAGED OBJECT CLASS**

| Object Name           | Object Type   | A | B | C | How Supported By RIC | How Supported By LERIC  |
|-----------------------|---------------|---|---|---|----------------------|-------------------------|
| <b>Hub Attributes</b> |               |   |   |   |                      |                         |
| hubID                 | ATTRIBUTE GET | X |   |   | Software (Note 1)    | Software                |
| hubGroupCapacity      | ATTRIBUTE GET | X |   |   | Software             | Software                |
| groupMap              | ATTRIBUTE GET | X |   |   | Software             | Software                |
| hubHealthState        | ATTRIBUTE GET | X |   |   | Software             | Software                |
| hubHealthText         | ATTRIBUTE GET | X |   |   | Software             | Software                |
| hubHealthData         | ATTRIBUTE GET | X |   |   | Software             | Software                |
| transmitCollisions    | ATTRIBUTE GET |   | X |   | TXCOL on Mngmt Bus   | External Logic on ANYXN |
| repeaterMJLPs         | ATTRIBUTE GET |   | X |   | JAB bit on Mngmt Bus | LEDs                    |

**Hub Actions**

|                        |        |   |  |  |          |          |
|------------------------|--------|---|--|--|----------|----------|
| resetHubAction         | ACTION | X |  |  | Software | Software |
| executeSelfTest1Action | ACTION | X |  |  | Software | Software |
| executeSelfTest2Action | ACTION | X |  |  | Software | Software |

**Hub Notifications**

|                |              |   |  |  |          |          |
|----------------|--------------|---|--|--|----------|----------|
| hubHealth      | NOTIFICATION | X |  |  | Software | Software |
| hubReset       | NOTIFICATION | x |  |  | Software | Software |
| groupMapChange | NOTIFICATION | X |  |  | Software | Software |

**ResourceTypeID Managed Object Class**

|                |  |   |  |  |          |          |
|----------------|--|---|--|--|----------|----------|
| resourceTypeID |  | X |  |  | Software | Software |
|----------------|--|---|--|--|----------|----------|

**Note 1:** In the "How Supported..." columns, when Software is noted, this means that the Object is independent of hardware, and is an object that is collected and maintained by software or ROM firmware.

**GROUP MANAGED OBJECT CLASS**

| Object Name             | Object Type   | A | B | C | How Supported By RIC | How Supported By LERIC |
|-------------------------|---------------|---|---|---|----------------------|------------------------|
| <b>Group Attributes</b> |               |   |   |   |                      |                        |
| groupID                 | ATTRIBUTE GET | X |   |   | Software             | Software               |
| numberOfPorts           | ATTRIBUTE GET | X |   |   | Software             | Software               |

## APPENDIX A. IEEE 802.3 HUB MANAGEMENT IMPLEMENTATION (Continued)

## PORT MANAGED OBJECT CLASS

| Object Name               | Object Type   | A | B | C | How Supported By RIC                         | How Supported By LERIC                   |
|---------------------------|---------------|---|---|---|----------------------------------------------|------------------------------------------|
| <b>Port Attributes</b>    |               |   |   |   |                                              |                                          |
| portID                    | ATTRIBUTE GET | X |   |   | Software                                     | Software                                 |
| portAdminState            | ATTRIBUTE GET | X |   |   | Ports' Real Time Status Register             | Ports' Real Time Status Register         |
| autoPartitionState        | ATTRIBUTE GET |   | X |   | Ports' Real Time Status Register             | Ports' Real Time Status Register         |
| readableFrames            | ATTRIBUTE GET |   | X |   | # of CLN bit active on Mngmt bus (no errors) | —                                        |
| readableOctets            | ATTRIBUTE GET |   | X |   | total # of RBY counts on Mngmt bus           | —                                        |
| frameCheckSequence-Errors | ATTRIBUTE GET |   | X |   | CRC bit active on Mngmt bus                  | —                                        |
| alignmentErrors           | ATTRIBUTE GET |   | X |   | FAE bit active on Mngmt bus                  | —                                        |
| framesTooLong             | ATTRIBUTE GET |   | X |   | RBY count on Mngmt bus                       | —                                        |
| shortEvents               | ATTRIBUTE GET |   | X |   | SE bit on Mngmt bus (no COL)                 | —                                        |
| runts                     | ATTRIBUTE GET |   | X |   | RBY count on Mngmt bus (no SE)               | —                                        |
| collisions                | ATTRIBUTE GET |   | X |   | Port Event Counters                          | External Logic Using DFS and LED Drivers |
| lateCollisions            | ATTRIBUTE GET |   | X |   | Event Logging Interrupts                     | —                                        |
| dataRateMismatches        | ATTRIBUTE GET |   | X |   | ELBER bit on Mngmt bus (no COL)              | —                                        |
| autoPartitions            | ATTRIBUTE GET |   | X |   | Event Logging Interrupts                     | Ports' Real Time Status Register         |
| lastSourceAddress         | ATTRIBUTE GET |   |   | X | Source address from Mngmt bus                | Source Address from Ext. Controller      |
| sourceAddressChanges      | ATTRIBUTE GET |   |   | X | Software                                     | Software                                 |
| <b>Port Actions</b>       |               |   |   |   |                                              |                                          |
| portAdminControl          | ACTION        | X |   |   | Ports' Real Time Status Register             | Ports' Real Time Status Register         |



# DP83956EB-AT LERIC™ (Lite Repeater Interface Controller) PC-AT® Adapter

National Semiconductor  
Application Note 854  
Marc Clevenger  
Imad Ayoub  
C.S. Balasvbramanian



## 1.0 INTRODUCTION

This LERIC-NIC Evaluation Board provides IBM® PC-AT and AT compatible computers with Twisted Pair conductivity. The board uses the DP8390 (NIC) to perform the Ethernet® protocol operations and the DMA operations. The dual DMA (local and remote) capabilities of the NIC, along with 16 kBytes of buffer RAM, allow the entire Network Interface Adapter to appear as a standard I/O Port to the system. The NIC module's local DMA channel buffers packets between the local memory (16 kBytes of buffer RAM) and the network, while the NIC module's remote DMA channel passes data between the local memory and the system memory by way of an I/O Port. This I/O Port architecture, which isolates the CPU from the network traffic, proves to be the simplest method to interface the DP8390 to the system. The DP83956 (LERIC) is used to interface to twisted pair Ethernet and provides IEEE 802.3 (Chapter 9) compliant repeater functions to six twisted pair ports. The LERIC has an on-chip PLL for Manchester data decoding, a Manchester encoder and an Elasticity buffer for preamble regeneration. It also has 6 integrated 10BASE-T transceivers. The LERIC's internal registers can be accessed using the same I/O port architecture as the NIC. This board provides the required attributes for compliance with Novell's® Hub Management Interface (HMI) basic control capability.

## 2.0 BOARD OVERVIEW

The LERIC-NIC board allows direct connection to the network using the RJ-45 phone jacks. There are 6 ports on a card. In addition, up to 4 boards can be cascaded together in a PC-AT, thus providing 24 Twisted Pair ports.

The block diagram shown in *Figure 1* illustrates the architecture of the LERIC-NIC Evaluation Board. The LERIC-NIC Board as seen by the PC-AT system appears only to be an I/O port. With this architecture the LERIC-NIC board has its own local bus to access the board memory. The system never has to intrude further than the I/O ports for any packet data operation.

### 2.1 Hardware Features

- Utilizes DP83956 Lite Repeater Interface Controller (LERIC)
- Six 10BASE-T connections per card and one node connection utilizing the NIC
- Cascadability of up to 4 boards
- 16 kByte on-board Packet Buffer
- Simple I/O Port Interface to IBM PC-AT
- Interfaces to Twisted Pair (10BASE-T)
- Boot EPROM Socket

The detailed schematics for this design are shown at the end of this document.

## 3.0 BOARD ARCHITECTURE

### 3.1 Board I/O Map

The LERIC-NIC Board requires a 32-byte I/O space to allow for decoding the data buffers, the reset port, and the NIC and LERIC registers. The first 16 bytes (300h–30Fh) are used to address the LERIC (4 bits wide) and NIC registers (8 bits wide) and the next 8 bytes (310h–317h) are used to address the data buffers which are 16 bits wide. Finally, the reset port (also software selectable) may be addressed by 318h–31Fh.

TABLE I. I/O Map in PC-AT

| Address  | Part Addressed   |
|----------|------------------|
| 300h–30f | NIC/LERIC Select |
| 310h–317 | Data Buffers     |
| 318h–31f | Reset            |

Although in the description above the I/O map is positioned at the addresses 300–31F, it may also be placed in the following address spaces: 320–33F, 340–35F, 360–37F. These alternate address spaces may be selected by the two jumpers (JP1 and JP0) as shown in Table II.

TABLE II. Optional Address Spaces

| JP1 | JP0 | I/O Address Space |
|-----|-----|-------------------|
| ON  | ON  | 300h–31Fh         |
| ON  | OFF | 320h–33Fh         |
| OFF | ON  | 340h–35Fh         |
| OFF | OFF | 360h–37Fh         |

### 3.2 Data and Address Paths

The following paragraph may be better understood by looking at the block diagram shown in *Figure 1*. Twenty address lines from the PC® go onto the LERIC-NIC Board, but only four of them actually go to the LERIC and the NIC. These four addresses along with the  $\overline{IOR}$  (low-asserted I/O read) or  $\overline{IOW}$  (low-asserted I/O write) and the  $\overline{CS}$  (NIC chip select signal) allow the PC to read or write to the LERIC and NIC's registers. If the system wants to read from or write to the LERIC or NIC registers, the data (8 bits for the NIC and 4 bits for the LERIC) must pass through the appropriate 245 buffer.

All of the packet data will pass through the I/O ports (the 374's). Each 374 is unidirectional and can only drive 8 bits, therefore it is necessary to have four 374's. Two of which drive data from the ports to the board memory and two of which drive the data from the ports to the AT bus. Even the PROM, which can only be addressed by the NIC, sends its 8 bits of data out through the 374's. When the PROM does this, two of the 374's will be enabled but only the lower 8 bits will have valid data. The RAM is also accessed by the NIC. However, it is addressed by 14 bits and drives out 16 bits of data.

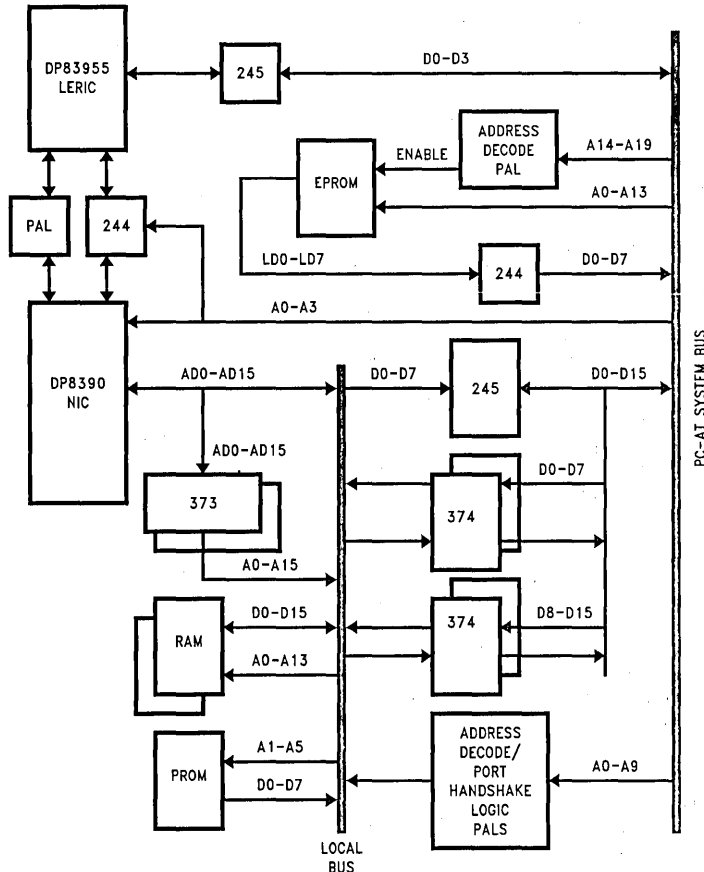


FIGURE 1. LERIC-NIC PC-AT System Interface

TL/F/11706-1

The PALs® receive 7 address lines among many other signals such as  $\overline{IOR}$ ,  $\overline{IOW}$ ,  $\overline{ACK}$ ,  $\overline{MRD}$ , etc. With these signals the PALs do all of the decodes, such as selecting the LERIC-NIC Board, the LERIC chip, the NIC chip, the RAM, and the PROM.

The EPROM socket is provided so that the user may add an EPROM to the system. This EPROM would normally contain a program and a driver to enable the PC-AT to be booted up through the network. The chips necessary to interface the EPROM to the system are the 27128 (EPROM), a 16L8 (PAL), and a 74ALS244 (buffer). Also, JP7 must be placed in the proper selection as described in the jumper section. The PAL decodes SA14-SA19, along with SMRDC (system memory read), in order to generate the EPROMEN signal. This signal, issued when the PC wants to execute the EPROM and the 244 buffer.

**3.3 Global Register Description**

An additional 3-bit write only register is provided on the board to allow for accesses to the LERIC. This register is also accessed using the I/O port architecture, and one address location in the I/O space has been allocated for it. This register can be mapped to one of four address location. This is done by using jumpers JP2 and JP3 as shown in Table III. The CPU can only write to this register.

TABLE III. Optional Address Spaces for Global Register

| JP2 | JP3 | I/O Address Space |
|-----|-----|-------------------|
| ON  | ON  | 200h              |
| OFF | ON  | 220h              |
| ON  | OFF | 240h              |
| OFF | OFF | 260h              |

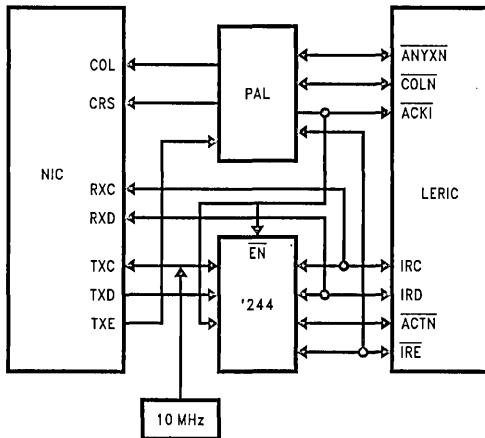
The three bits in this register are shown in Table IV. Bits 0 and 1 are used to specify the board number. Since there can be up to four boards cascaded together, a unique board address is necessary to distinguish between them. Bit 2 is used to select the NIC or the LERIC when CPU accesses are being made. On power-up this bit defaults to one and selects the NIC. In order to access the LERIC, this bit has to be set to zero.

TABLE IV. Global Register Bits

| Bit 2     | Bit 1   | Bit 0   |
|-----------|---------|---------|
| NIC/LERIC | Board # | Board # |

## 4.0 LERIC-NIC INTERFACE

The LERIC-NIC interface makes use of the Inter-LERIC™ Bus which consists of the following signals:  $\overline{ACKO}$ ,  $\overline{ACKI}$ , IRC, IRD,  $\overline{IRE}$ ,  $\overline{ACTN}$ ,  $\overline{ANYXN}$  and  $\overline{COLN}$ . Besides NRZ data (IRD) and clock (IRC) this bus provides other signals necessary for cascading one LERIC to another. The NIC is treated as another LERIC when it is connected to the Inter-LERIC bus. The Inter-LERIC Bus also eliminates the need for an encoder/decoder chip to which the NIC is usually connected. Since the Inter-LERIC Bus is bidirectional and some logical operations are necessary to convert the signals to be compatible with the NIC, a PAL and some TRI-STATE® buffers are used to implement this function. Figure 2 shows the interface between the LERIC and NIC. In this implementation the NIC is placed on the top of the arbitration chain. The NIC input pins RXC and RXD are connected directly to the LERIC pins IRC and IRD, respectively. The  $\overline{COL}$  input of the NIC is derived by combining the  $\overline{COLN}$  and  $\overline{ANYXN}$  signals from the LERIC. The CRS input on the NIC comes from the inverted  $\overline{IRE}$  signal of the LERIC. When the NIC wants to transmit, it drives the  $\overline{ACKI}$  input of the LERIC with the inverted TXE signal. The inverted TXE signal is also used to enable the 244 TRI-STATE buffer which connects the NIC output signals TXD, TXC to the IRD and IRC signals on the Inter-LERIC Bus.  $\overline{TXE}$  is used to drive the  $\overline{ACTN}$  and  $\overline{IRE}$  signals during transmission.



TL/F11706-2

CRS =  $\overline{IRE}$   
 $\overline{COL} = \overline{COLN} + \overline{ANYXN}$   
 $\overline{EN} = \overline{TXE}$

FIGURE 2. LERIC-NIC Interface

### 4.1 Inter-LERIC Bus Interface

The LERIC-NIC board, or master board, may also be used without the NIC. A board with the LERIC only (no NIC), or slave board, is desirable when more boards are to be used in the same PC. One master and up to three slave boards can be cascaded using the Inter-LERIC Bus interface to form a larger logical repeater, (i.e., one that meets IEEE's specification for a single repeater). There are two 14-pin headers on the board for cascading the Inter-LERIC bus signals. Both headers contain the signals IRC, IRD,  $\overline{IRE}$ ,  $\overline{ACTN}$ ,  $\overline{ANYXN}$ , and  $\overline{COLN}$ . The input header, J7, also contains the  $\overline{ACKI}$  signal, and the output header, J9, contains  $\overline{ACKO}$ . These signals enable multiple LERICs to be cascaded

ed together. The  $\overline{ACKI}$  and  $\overline{ACKO}$  signals are daisy chained between the boards. The  $\overline{ACKO}$  signal will drive the  $\overline{ACKI}$  input of the board which is next on the arbitration chain. Jumper JP8 (see schematic) is used to tie  $\overline{ACKI}$  to the inverted TXE signal from the NIC, which puts the NIC at the top of the arbitration chain. If all boards are used in slave mode, JP8 can also tie  $\overline{ACKI}$  high, putting that board on top of the chain. Otherwise, with JP8 removed, the  $\overline{ACKI}$  signal will be driven by the  $\overline{ACKO}$  output of the board higher up on the arbitration chain. Since these signals are held TRI-STATE or open collector they are pulled up by resistors. The resistor value of 8.2 k $\Omega$  is selected for these pull-ups. When all four boards are cascaded, the smallest pull-up value on any Inter-LERIC signal will be approximately 2 k $\Omega$ . This elevates the need to remove some of these pull-up resistors when additional boards are cascaded.

## 5.0 BOARD OPERATION

The following pages describe the slave accesses to the LERIC-NIC and the local DMA and remote DMA operation.

### 5.1 Global Register Operations

Accesses to the board are register operations to the NIC or the LERIC, which are done to set up the NIC to control the operation of the NIC's DMA channels, and read and write to the LERIC registers. Since the NIC and LERIC share the same I/O space for the registers (300h-30Fh), an additional CPU operation is required. Before any register read or write, the CPU performs a write to the global register bit 2 in order to select the LERIC or NIC and to bits 0 and 1 selecting one of the four possible boards. The usage of this bit depends on the software used. If a normal network (no hub access) driver is used the card looks like a pure adaptor to the software. The board normally would be set with the NIC selected. When the driver needs to access the LERIC it would first write to the Global Register, do the LERIC operations, then set it back to enable the NIC access. This minimizes the changes to the NIC portion of the driver.

To begin the global register write (see Section 3.0 for the Global Register description), the CPU drives the SA0-SA3 address lines to the LERIC-NIC board and the SA4-SA9 address lines to the PAL. With these address lines, the PAL decodes to 200, 220, 240 or 260 depending on the settings of jumpers JP2 and JP3. The CPU then drives the  $\overline{IOW}$  strobe which is used to latch in the data on the AT bus into the Global Register on the rising edge of  $\overline{IOW}$ . This ends the cycle of the global register write.

### 5.2 LERIC Register Accesses

Before any LERIC register access, the CPU must write to the global register in order to select the LERIC and the appropriate LERIC-NIC board. After the register access, the CPU must perform another write to the global register to select the NIC.

#### 5.2.1 LERIC Register Read

To begin the register read, the CPU drives the four address lines (SA0-SA3) to the LERIC and the SA4-SA9 address lines to the PAL. With these address lines and the  $\overline{IOR}$  line, the PAL decodes to 300-30F (the LERIC registers) and the  $\overline{LRD}$  signal is enabled. Once the LERIC receives this  $\overline{LRD}$ , it then sends out a low assertion on  $\overline{BUFEN}$ . The  $\overline{BUFEN}$  signal is used by the PAL to assert the IOCHRDY signal false. The LERIC then drives out the data from its internal registers to the 245 buffer. The 245 buffer is then enabled by the  $\overline{LERICEN}$  signal and the data is driven onto the AT BUS. A 3-bit counter is used to indicate when the LERIC has

driven the data out. This gives the LERIC enough time to output the data. This is required since the LERIC does not have any other signal which indicates that the data is available. This is indicated by the signal COUNT<sub>4</sub> going high (after about 400 ns) which causes the PAL to assert IOCHRDY true. As a result,  $\overline{IOR}$  is driven high by the CPU, thereby de-asserting the  $\overline{LRD}$ . On the rising edge of the  $\overline{IOR}$ , the data which is on the AT BUS is latched into the system. The addresses are removed at the same time, causing the LERIC chip select to become de-asserted, ending the register read cycle.

### 5.2.2 LERIC Register Write

To begin the register write, the CPU drives the four address lines (SA0-SA3) to the LERIC and the SA4-SA9 address lines to the PAL. With these address lines and the  $\overline{IOW}$  line, the PAL decodes to 300-30F (the LERIC registers) and the  $\overline{LWR}$  signal is enabled. Once the LERIC receives this  $\overline{LWR}$ , it then sends out a low assertion on  $\overline{BUFEN}$ . The  $\overline{BUFEN}$  signal is used by the PAL to assert the IOCHRDY (used to insert wait states) signal false. The CPU then drives out the data onto the AT BUS where it goes into the 245 buffer. The 245 buffer is then enabled by the  $\overline{LERICEN}$  signal and the data is driven to the LERIC. A 3-bit counter is used to indicate when the LERIC has latched the data in. This is indicated by the signal COUNT<sub>4</sub> going high which causes the PAL to assert IOCHRDY true. As a result,  $\overline{IOW}$  is driven high by the CPU, thereby de-asserting the  $\overline{LWR}$ . The addresses are removed at the same time, causing the LERIC chip select to become de-asserted, ending the register write cycle.

### 5.3 NIC Register Accesses

The following discussion assumes a jumper setting for I/O address space of 300h-31Fh.

#### 5.3.1 NIC Register Read

To begin the register read, the CPU drives the four address lines (SA0-SA3) to the NIC and the SA4-SA9 address lines to the PAL. With these address lines, PAL #2 decodes to 300-30F (the NIC registers) thereby enabling the chip select for the NIC. The CPU also drives the  $\overline{IOR}$  line which the NIC sees as the  $\overline{SRD}$  (slave read). Since the NIC may be a local bus master when the CPU attempts to read or write the controller, an  $\overline{ACK}$  line is used by the PAL to assert the IOCHRDY signal false and wait state the CPU. The NIC drives out the data from its internal registers to the 245 buffer. When the NIC is ready to be in slave mode and complete the read cycle, it asserts  $\overline{ACK}$  true which enables the 245 buffer and the data is driven onto the AT BUS. Driving  $\overline{ACK}$  true also causes the PAL to assert IOCHRDY true. As a result,  $\overline{IOR}$  is driven high by the CPU, thereby de-asserting the  $\overline{SRD}$ . On the rising edge of the  $\overline{IOR}$ , the data which is on the AT BUS is latched into the system. The addresses are removed at the same time, causing the NIC chip select to become de-asserted, ending the register read cycle.

#### 5.3.2 NIC Register Write

To begin the register write, the CPU drives the SA0-SA3 address lines to the NIC and the SA4-SA9 address lines to the PAL. With these address lines, the PAL decodes to 300-30F (the NIC registers) thereby enabling the chip select for the NIC. The CPU then drives the  $\overline{IOW}$  strobe which the NIC sees as  $\overline{SWR}$  (slave write). Once the NIC receives this  $\overline{SWR}$  it sends back a low assertion on  $\overline{ACK}$  to acknowledge that it is in slave mode and ready to perform the write. A low assertion on  $\overline{ACK}$  will generate IOCHRDY true and enable the 245 buffer. The 245 buffer then drives the data from the AT BUS to the NIC. The system drives  $\overline{IOW}$  high, thereby de-asserting the  $\overline{SWR}$  and latching the data. The

addresses also are taken away and the chip select then goes high (de-asserted). This ends the cycle of the register write.

### 5.4 NIC Local Memory Map

There are only two items mapped into the local memory space. These two items, shown in Table V, are the 8k x 16k buffer RAM and the ID address PROM. The buffer RAM is used for temporary storage of transmit and receive packets. For transmit packets, the remote DMA puts data from the I/O ports into the RAM and the local DMA moves the data from the RAM to the NIC. For the receive packets, the local DMA carries the data from the NIC to the RAM and the remote DMA moves the data from the RAM to the I/O ports.

TABLE V. NIC Local Memory Map

|       |      |
|-------|------|
| 7FFFh | RAM  |
| 4000h |      |
| 3FFFh | PROM |
| 0000h |      |

The ID address PROM (74S288 32 x 8) contains the physical address of the evaluation board. Each PROM holds its own unique physical address which is installed during its manufacture. Besides this address, the PROM also contains a checksum. This checksum, calculated by exclusive OR-ing the six address bytes with each other, is provided in order to check the addresses. At the initialization of the evaluation board the software commands the NIC to transfer the PROM data to the I/O Port where it is read by the CPU. The CPU then verifies the checksum and loads the NIC's physical address registers. Table VI shows the contents of the PROM.

TABLE VI. PROM Contents

| PROM Location | Location Contents                          |
|---------------|--------------------------------------------|
| 00h           | Ethernet Address 0 (most significant byte) |
| 01h           | Ethernet Address 1                         |
| 02h           | Ethernet Address 2                         |
| 03h           | Ethernet Address 3                         |
| 04h           | Ethernet Address 4                         |
| 05h           | Ethernet Address 5                         |
| 06h-0Dh       | 00h                                        |
| 0Eh, 0Fh      | 57h                                        |
| 10h-15h       | Ethernet Address 0 through 5               |
| 16h-1Dh       | Reserved                                   |
| 1Eh, 1Fh      | 42h                                        |

### 5.5 NIC Remote DMA Packet Data Transfers

Remote DMA transfers are operations performed by the NIC on the board. These operations occur when the NIC is programmed to transfer packet data between the PC-AT and the card's on-board RAM. These transfers take place through the I/O Port interfacing.

### 5.5.1 Remote Read

To program the NIC for a remote read, the CPU must take five slave accesses to the NIC. The CPU must write the Remote Start Address (2 bytes) and the Remote Byte Count (2 bytes). Then the CPU issues the Remote DMA Read command.

Once the NIC has received all of the above data, it drives out BREQ and waits for BACK. The NIC immediately receives BACK because it is tied to the BREQ line. BREQ can be tied to BACK because there are no other devices contending for the local bus. After receiving the BACK, the NIC drives out the address from which the data is required to be read. This address flows into the 373's and is latched by ADS0. From here, the address flows to the RAM. The RAM waits until it receives MRD from the NIC and then it drives the data into the 374 ports. The 374 ports then latch the data on the rising edge of the PWR strobe from the NIC. PRQ is then sent out by the NIC to let the system know that there is data waiting in the ports.

If the AT reads the I/O ports before the NIC has loaded the 374's, then the port request (PRQ) from the NIC will not yet be driven. This unasserted PRQ signal causes the AT's ready line to be set low, indicating that the NIC has yet to load the data. After the data is in the ports, the system must then read the 374 data ports. This begins with the AT driving out an address which is decoded (inside PAL #1) to the data I/O Ports (310-317). PAL #2 then drives RACK to the NIC, indicating that the CPU is ready to accept data. This RACK signal then reads the data from the 374 ports onto the AT BUS. The system deasserts IOR which finishes the cycle.

### 5.5.2 Remote Write

Like the remote read, the remote write cycle also begins with five slave accesses to the internal registers. The CPU must write the Remote Start Address (2 bytes), the Remote Bytes Count (2 bytes), and issue the Remote DMA write command. The NIC then issues a PRQ. The CPU responds by sending an IOW, indicating that it is ready to write to the ports. The CPU also drives out the address which corresponds to the I/O Ports. PAL #2 generates WACK on an address decode to the data buffers along with PRQ and IOW. This WACK signal latches the data into the 374 ports. The NIC issues a BREQ and immediately receives BACK since the two lines are tied together. The NIC, upon receiving BACK, drives out address lines to the 373's. These address lines are latched by ADS0 and then are driven to the RAM. The NIC then sends out a PRD and a MWR which drives the data from the 374 ports into the already specified address of the onboard RAM. PRD and MWR are then deasserted and the cycle ends.

### 5.6 Network Transfers from NIC to Buffer RAM

Transfers to and from the network are controlled by the NIC's local DMA channel which transfers packet data to/from the NIC's internal FIFO from/to the card's buffer RAM.

#### 5.6.1 Data Reception

The data received from the network, is deserialized and is loaded into the FIFO inside of the NIC. The NIC then issues a BREQ and immediately receives BACK since the lines are tied together. After receiving BACK, the NIC drives the address lines to the 373's. The 373's are latched by ADS0 and the address is allowed to flow to the RAM. Then the NIC drives out MWR along with the data from the FIFO. The data flows into the RAM at the address given earlier. The MWR strobe is then deasserted, ending the cycle.

### 5.6.2 Data Transmision

To begin the transmit cycle, the NIC issues a BREQ and waits for BACK. Since BREQ and BACK lines are tied together, BACK is received immediately. Upon reception of this signal, the NIC drives out the address to the 373's which latch the address with the ADS0 strobe. The address then flows to the on-board memory. MRD, driven by the NIC, causes the RAM to drive the data out of the given address and into the NIC. The NIC then latches the data into the FIFO on the rising edge of MRD. The high assertion of MRD signifies the ending of this cycle. From the FIFO, the data is serialized and transmitted on the network.

### 6.0 TWISTED PAIR INTERFACE

The interface to the network through the LERIC twisted pair ports is shown in *Figure 3*. To drive the transmitted signal through 100 meters of Unshielded Twisted Pair (UTP) cable, the LERIC requires external drivers. The optimized resistor network shown provides the proper pre-emphasis on the transmit signals and the 100Ω termination on the receive pair. Standard Filter Transformer Choke modules (such as Valer FL1012) are used to provide the required filtering and isolation.

### 7.0 BOARD CONFIGURATION

The LERIC is initialized during power-on reset. On the rising edge of the reset signal from the PC-AT, the data on the pins D0-D7 are loaded into the configuration registers. This reset is tied directly to the MLOAD pin of the LERIC. (Refer to the LERIC datasheet for the description of MLOAD.) On the LERIC-NIC board there are pull-up resistors on pins D0-D7, which will load 1's into the configuration registers during power up. The all 1's pattern configures the LERIC in the six twisted pair ports and one full AUI mode, enables the polarity switching on the twisted pair ports, selects the 31 consecutive collision counts on a port before partition and sets the LED update operation in the maximum mode.

The LERIC possesses control logic and interface pins which may be used to provide status information concerning activity on the attached network segments and the current status of repeater functions. On the LERIC-NIC board, 7 LED's are provided (one for each port and one for updating "any" port status). A '259 addressable latch is used to latch the data and address information contained in the D(7:0) pins of the LERIC (refer to the LERIC datasheet for the description of each pin). A toggle switch, SW1, is also provided to allow the display of either the status of the link integrity or reception activity on a per port basis and the status of collisions or jabber on the "any" port LED. This switch selects which LERIC data pin to use as data input to the '259 latch.

An on-board crystal oscillator provides the clock inputs for the NIC and LERIC. The oscillator's output is 20 MHz, which is fed directly to the LERIC and the NIC. Since the NIC also requires a 10 MHz clock, a flip-flop (74ALS74) is used to divide the 20 MHz down to 10 MHz. This is fed to the TXC input of the NIC.

The 10 MHz clock is also used as an input to the 74LS93 counter. Since there is no "Ready" signal from the LERIC indicating the completion of the data latching on a register read or write operation, this counter is used to generate a wait state before the I/OCHRDY signal is asserted back to the CPU.

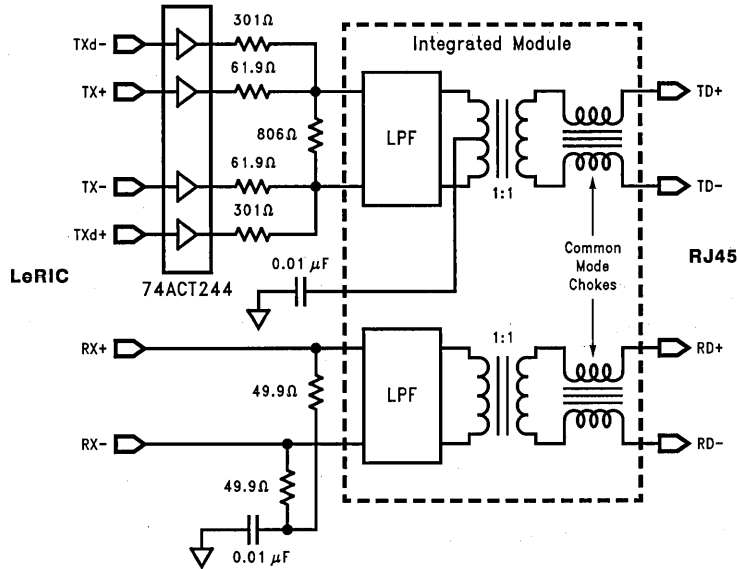


FIGURE 3. Twisted Pair Interface

TL/F/11706-3

7.1 Jumper Options

The default jumper block configurations are shown in Figure 4. On JB4, there are six possible connections. Four of these are to select an interrupt line. The available interrupt lines include INT3, INT4, INT5, and INT9. The last two possible connections, JP1 and JP0, are used to select the base address for the board. However, if JP7 is connected to V<sub>CC</sub>, then these last two connections also select the address of the EPROM.

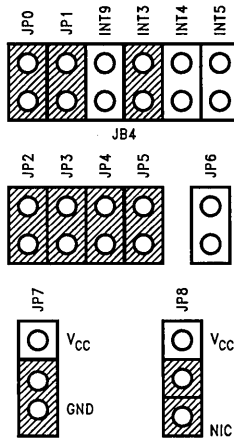


FIGURE 4. Jumper Blocks

TL/F/11706-4

The possible selections and the jumpers which should be ON (closed) are shown in Table VII. The factory configuration uses the INT3 line for interrupts and has JP1 and JP0 in the ON position. JP2 and JP3 are used to set the optional address space for the global register. The default position is ON for these jumpers (see Table III).

TABLE VII. Base and EPROM Addresses

| JP1 | JP0 | Base Add  | EPROM Add |
|-----|-----|-----------|-----------|
| ON  | ON  | 300h-31Fh | C800h     |
| ON  | OFF | 320h-33Fh | CC00h     |
| OFF | ON  | 340h-35Fh | D000h     |
| OFF | OFF | 360h-37Fh | D400h     |

JP4 and JP5 are used to set the board address when multiple boards are cascaded together. These jumpers are normally ON in the default position. JP6 is used to take care of the IOCHRDY timing issues in some clone ATs. The default for this jumper is OFF (refer to PAL # 2 description for more details on the IOCHRDY timing inconsistency). JP8 is used for arbitration when cascading several boards. The default position assumes that the NIC is on top of the arbitration chain, and ACKI is generated from PAL # 4. When cascading several boards, the boards lower on the chain would have JP8 removed.

**8.0 PAL EQUATIONS****PAL # (U1)**

In this PAL, the output signals are NIO16, NIOEN, NNICB and SELREG.

NIO16 is only asserted if the CPU is trying to access the local RAM buffers and the board is in a 16-bit slot. NIO16 is used by the CPU to determine if it should perform 8- or 16-bit operations. If NIO16 is false, then the CPU will only perform 8-bit operations. Since it is necessary to assert NIO16 as soon as possible, this PAL has been selected to be a 10 ns "D" PAL. The NIO16 signal must be TRI-STATE when it is not asserted. Therefore, we use an enable signal (NIOEN) which is equal to the decode for the I/O Ports

(310-31F) and NAEN high (NAEN high signifies that the system DMA does not have control of the bus). The enable signal (NIOEN) loops back into the PAL to bring NIO16 out of TRI-STATE. The NIO16 signal is set to zero so that whenever it is enabled it will be asserted.

The NNICB signal consists of simple address decodes along with NAEN, and will be asserted when the CPU wants to access the LERIC-NIC board. The addresses decode to one of four address slots which were earlier mentioned in the board configuration section.

The SELREG signal is similar to NNICB and decodes to one of four addresses which were mentioned in the board architecture section. SELREG is asserted when the CPU wants to access the Global Register.

**PAL 1**

```
begin header
```

```
date: 4/2/91
```

```
functions:
```

```
NIC/LERIC BOARD DECODE, IO16 DECODE, AND GLOBAL REGISTER DECODE';
```

```
end header
```

```
begin definition
```

```
device gall16v8;
```

```
inputs
```

```
  NEN16=1, NAEN=2, SA9=3,
```

```
  SA8=4, SA7=5, SA6=6,
```

```
  SA5=7, SA4=8, SA3=9,
```

```
  JP0=13, JP1=14, JP2=15, JP3=16;
```

```
outputs (com)
```

```
  /NNICB=12, /NIO16=18, SELREG=19;
```

```
feedbacks (com)
```

```
  NIOEN=17;
```

```
end definition
```

```
begin equations
```

```
NNICB =      (!NAEN & SA9 & SA8 & !SA7 & !SA6 & !SA5 & !JP1 & !JP0
# !NAEN & SA9 & SA8 & !SA7 & !SA6 & SA5 & !JP1 & JPO
# !NAEN & SA9 & SA8 & !SA7 & SA6 & !SA5 & JP1 & !JPO
# !NAEN & SA9 & SA8 & !SA7 & SA6 & SA5 & JP1 & JPO);
```

```
SELREG =     (!NAEN & SA9 & !SA8 & !SA7 & !SA6 & !SA5 & !JP3 & !JP2
# !NAEN & SA9 & !SA8 & !SA7 & !SA6 & SA5 & !JP3 & JP2
# !NAEN & SA9 & !SA8 & !SA7 & SA6 & !SA5 & JP3 & !JP2
# !NAEN & SA9 & !SA8 & !SA7 & SA6 & SA5 & JP3 & JP2);
```

```
NIOEN =      (!NAEN & SA9 & SA8 & !SA7 & !SA6 & !SA5 & !JP1 & !JPO
& !NEN16 & SA4 & !SA3
# !NAEN & SA9 & SA8 & !SA7 & !SA6 & SA5 & !JP1 & JPO
& !NEN16 & SA4 & !SA3
# !NAEN & SA9 & SA8 & !SA7 & SA6 & !SA5 & JP1 & !JPO
& !NEN16 & SA4 & !SA3
# !NAEN & SA9 & SA8 & !SA7 & SA6 & SA5 & JP1 & JPO
& !NEN16 & SA4 & !SA3);
```

```
NIO16.oe = NIOEN;
```

```
NIO16 = 1;
```

```
end equations
```

**PAL # (U2)**

In this PAL, there are seven outputs which include NRESET, NIOCHR, NIOCHW, NIOCHC, NCS, NRACK, and NWACK.

The IOCHRDY signal is used to wait state the CPU. Normally an I/O card drives IOCHRDY low (not ready) only *after* the address and I/O read or write signals have been asserted. On some PC-AT compatible PCs (those using Chips and Technologies or VLSI Inc. chipsets), during a 16-bit I/O operation, the bus controller actually samples the IOCHRDY signal *before* the I/O read or write signal is asserted.

NIOCHC is used to drive IOCHRDY low (not ready) based only on an address decode, thus allowing IOCHRDY to be asserted earlier and in the proper state when it is sampled by the bus controller.

As shown in *Figure 5*, NIOCHR, NIOCHW, and NIOCHC are all externally wire-ORed together to generate the IOCHRDY signal. NIOCHR along with NIOCHR.OE (enable NIOCHR) are for slave read and remote read cycles. NIOCHW and NIOCHW.OE (enable NIOCHW) are for slave write and remote write cycles. These signals together generate the "normal" IOCHRDY signal.

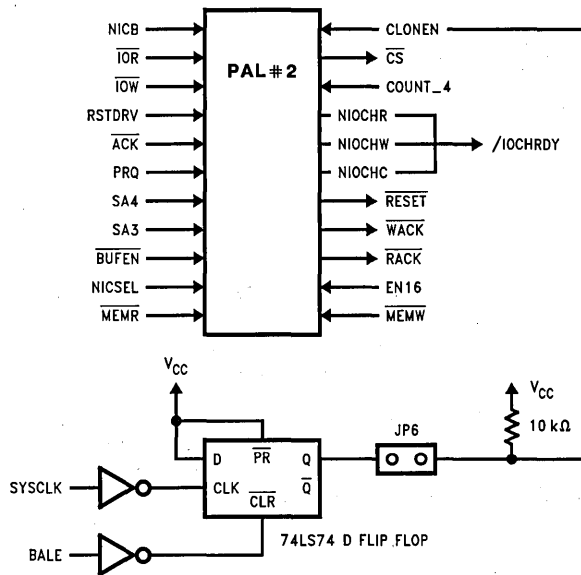
For more details on the IOCHRDY fix, refer to the application note *PC-AT Interface Design Considerations for the DP83902EB-AT*. NIOCHC may cause problems on PC's that

do not use Chips and Technologies or VLSI Inc. chipsets. A D-flip flop is used to generate CLONEN to implement the timing modification. The variable CLONEN is tied to a jumper that is used to switch the IOCHRDY signal characteristics to "normal" or modified timing. If the system operates under "normal" timing characteristics, JP6 should be removed.

NCS is asserted by the CPU when it needs to access the NIC registers. NCS is decoded in the address range of 300-30F along with NICSEL from PAL #5. NICSEL is set high to access NIC registers and low to access LERIC registers.

The next two signals, NRACK and NWACK, are used to acknowledge successful transfers between the CPU and the NIC data buffers. The NRACK occurs with an address decode to 310-317, an NIOR, and a PRQ. NWACK occurs with an address decode to 310-317, an NIOW, and a PRQ.

The last signal is NRESET, which is used to reset both the NIC and LERIC configuration registers. NRESET can be asserted by the system with RSTDRV or through software after the system has booted up. Once NRESET has been asserted, it is held low until IOW is received from the CPU. This will guarantee that NRESET has the pulse width required by the NIC.

**FIGURE 5**

TL/F/11706-6



PAL 2

begin header

date: 4/2/92

functions:

RESET LATCH, NIC SELECT, IOCHRDY, RACK, WACK ';

end header

begin definition

device PAL20L8;

inputs

NNICB=1, NIOW=2, NIOR=3,

RSTDRV=4, NACK=5, PRQ=6,

SA4=7, SA3=8, NBUFEN=9, NICSEL=10,

NMEMR=11, NMEMW=13, EN16=14, COUNT\_4=21, CLONEN=23;

outputs(com)

!NRACK=15, !NWACK=16,

!NIOCHR=18, !NIOCHW=19,

!NIOCHC=20, !NCS=22;

feedbacks(com)

!NRESET=17;

end definition

begin equations

```

NCS = (!NNICB & !NIOR & !SA4 & NICSEL
      # !NNICB & !NIOW & !SA4 & NICSEL);
NRACK = (!NNICB & PRQ & !NIOR & SA4 & !SA3 & NICSEL);
NWACK = (!NNICB & PRQ & !NIOW & SA4 & !SA3 & NICSEL);
NIOCHR= (!PRQ & SA4 & !SA3 & NICSEL
      # NACK & !SA4 & NICSEL
      # !NICSEL & !NBUFEN & !COUNT_4);
NIOCHR.oe= (!NNICB & !NIOR);
NIOCHW= (!PRQ & SA4 & !SA3 & NICSEL
      # NACK & !SA4 & NICSEL
      # !NICSEL & !NBUFEN & !COUNT_4);
NIOCHW.oe= (!NNICB & !NIOW);
NIOCHC=1;
NIOCHC.oe= (!NNICB & NMEMR & NMEMW & !PRQ & !EN16 &
          !CLONEN & SA4 & !SA3);
NRESET = (!NNICB & !NIOR & SA4 & SA3 # NIOW & NRESET # RSTDRV);

```

end equations

TL/F/11706-7

**PAL #3 (U16)**

The third PAL does a decode to enable the optional EPROM. This decode consists of an address decode to C8000h, CC000h, D0000h, or D4000h depending on JP1 and JP0 as shown in the board configuration section. JP7 must also be jumpered for selection of the EPROM. NAEN, a low asserted signal should be low to indicate that the DMA does not have control of the bus and the NSMRDC signal should be asserted low since the CPU is doing a system memory read.

**PAL 3**

```
begin header
date: 4/2/92
function:
EPROM DECODE';
end header
```

```
begin definition
```

```
device pal1618;
```

```
inputs
```

```
A0=1, A13=2, EN16=3,
NSMRDC=4, SA19=5, SA18=6,
SA17=7, SA16=8, SA15=9,
SA14=11, NAEN=13, JP2=14,
JP0=15, JP1=16;
```

```
outputs(com)
```

```
!NEPROMEN=19, !A013=12;
```

```
{Note: I/O pins 17 and 18 are not being used.}
```

```
end definition
```

```
begin equations
```

```
NEPROMEN = (SA19 & SA18 & !SA17 & !SA16 & SA15 & !SA14 & !NAEN
            & JP2 & !JP1 & !JP0 & !NSMRDC
            # SA19 & SA18 & !SA17 & !SA16 & SA15 & SA14 & !NAEN
            & JP2 & !JP1 & JP0 & !NSMRDC
            # SA19 & SA18 & !SA17 & SA16 & !SA15 & !SA14 & !NAEN
            & JP2 & JP1 & !JP0 & !NSMRDC
            # SA19 & SA18 & !SA17 & SA16 & !SA15 & SA14 & !NAEN
            & JP2 & JP1 & JP0 & !NSMRDC);
```

```
A013 = (!A0 & EN16 # !A13 & !EN16);
```

```
end equations
```

The A013 signal is used for 8-bit mode operation. When the board is placed into an 8-bit slot the EN16 signal is used to detect the existence of the second PC-AT bus connector. If EN16 is low then the board is in a 16-bit slot. In this condition, A13 (from the NIC) is enabled to A013 which goes to the LSB RAM. If the board is placed into an 8-bit slot then EN16 is pulled high by a resistor, and this causes the LER-IC-NIC's A0 signal to be enabled. This allows the LSB RAM to be used as an 8-bit RAM.

**PAL #4 (U30)**

In this PAL there are seven outputs which include CRS, COL, TXENABLE, NLRD, NLWR, NLERICEN, and GRDATA.

The first three outputs generate the interface signals between the LERIC and NIC. CRS is just an inverted NIRE signal which notifies the NIC that there is activity on the network, and COL is asserted if there is a receive or transmit collision coming from the LERIC. TXENABLE is tied to ACKI, putting the NIC at the top of the LERIC's arbitration chain. TXENABLE is also used to enable the 244 buffer that controls the flow of TXE, TXC, and TXD coming from the NIC.

NLRD, and NLWR generate the read and write strobes to the LERIC based on an address decode,  $\overline{\text{IOR}}$  or  $\overline{\text{IOW}}$ , and LERICIT. LERICIT is the Global Register decode from PAL #5.

The NLERICEN signal is used to enable the TRI-STATE buffers and receivers that access the LERIC registers. It will only be asserted after NLRD or NLWR are true and the LERIC has driven  $\overline{\text{BUFEN}}$  low, signifying that the inter-LE-ERIC BUS is available to do a register read or write.

The last output, GRDATA, is used to latch the Global Register bits into PAL #5. The data bits from the AT-BUS will be valid after  $\overline{\text{IOW}}$  is asserted low along with SELREG, which is an address decode for the Global Register coming from PAL #1.

**PAL 4**

```
begin header
```

```
date: 4/2/92
```

```
function:
```

```
LERIC READ, LERIC WRITE, LERIC MLOAD, CRS, Global Register Data;
```

```
end header
```

```
begin definition
```

```
device pal1618;
```

```
inputs
```

```
NIOR=1, NIOW=2, SELREG=3, LERICIT=4,  
NBUFEN=5, NANYXN=6, NCOLN=7, NIRE=8,  
SA4=9, NICB=11, TXE=16;
```

```
outputs(com)
```

```
!NLWR=12, !NLRD=13, !TXENABLE=14, !NLERICEN=15,  
!CRS=17, !COL=18, !GRDATA=19;
```

```
end definition
```

```
begin equations
```

```
CRS =          NIRE ;  
!COL =        !NCOLN # !NANYXN;
```

```
TXENABLE =     TXE;
```

```
NLRD = (!NICB & !NIOR & !SA4 & LERICIT ) ;  
NLWR = (!NICB & !NIOW & !SA4 & LERICIT ) ;
```

```
NLERICEN =     (!NICB & !NIOR & !SA4 & LERICIT & !NBUFEN  
               # !NICB & !NIOW & !SA4 & LERICIT & !NBUFEN);
```

```
!GRDATA =      SELREG & !NIOW;
```

```
end equations
```

TL/F/11706-9

**PAL #5 (U29)**

This GAL includes the outputs NCSROM, INTO, LERICHIT and NICSEL. The two registered outputs ID0 and ID1 are only used internally by the GAL.

The LERICHIT signal comes from the decode of the Global Register, and indicates that the CPU needs to access the LERIC.

**PAL 5**

```
begin header
date: 8/7/92
function:
LERICHIT, NICSEL, INTO, NCSROM';
end header
```

**begin definition**

```
device gal16v8;
```

**inputs**

```
GRDATA=1, JP4=2, JP5=3,
D2=4, D1=5, D0=6, NIOR=7,
A14=8, NMRD=9, INT=18;
```

**outputs(com)**

```
!NCSROM=12, INTO=13,
LERICHIT=19, NREAD=14;
```

**feedbacks(reg)**

```
ID0=15, ID1=16, NICSEL=17;
```

**end definition****begin equations**

```
INTO = INT;
NREAD= NIOR;
```

```
ID0 := D0;
ID1 := D1;
```

```
NICSEL := D2;
```

```
NCSROM = ( !A14 & !NMRD);
```

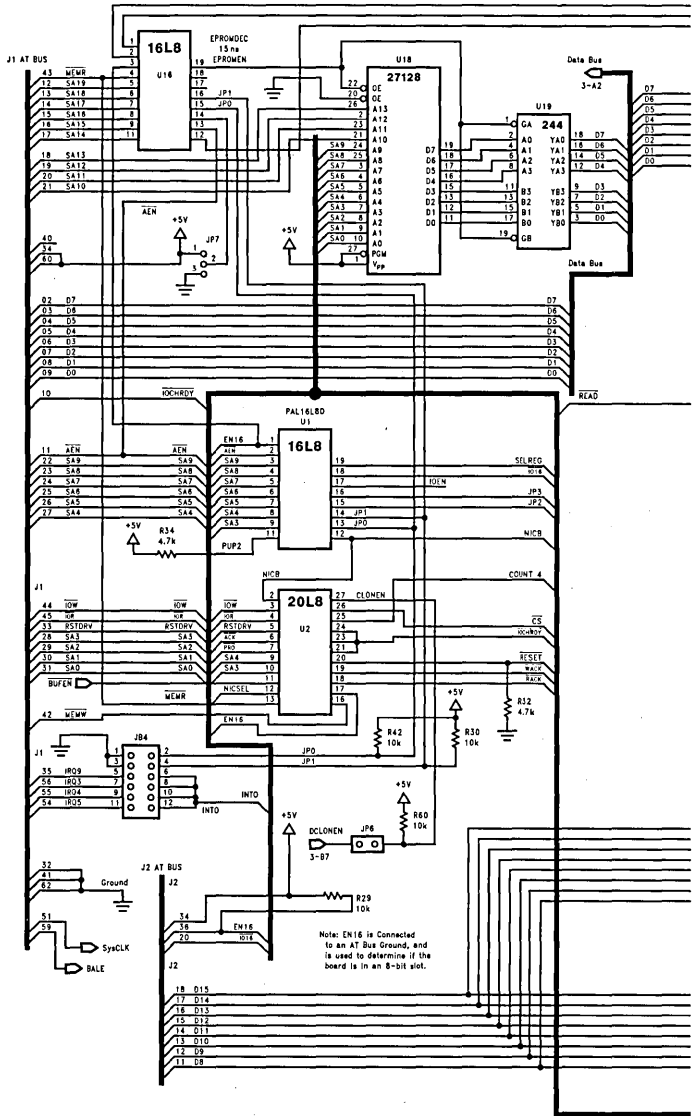
```
LERICHIT = (ID0 & ID1 & JP4 & JP5 & !NICSEL
# !ID0 & !JP4 & ID1 & JP5 & !NICSEL
# ID0 & JP4 & !ID1 & !JP5 & !NICSEL
# !ID0 & !JP4 & !ID1 & !JP5 & !NICSEL);
```

**end equations**

GRDATA is used to clock in the registered outputs consisting of ID0, ID1, and NICSEL. These three bits store the contents of the Global Register.

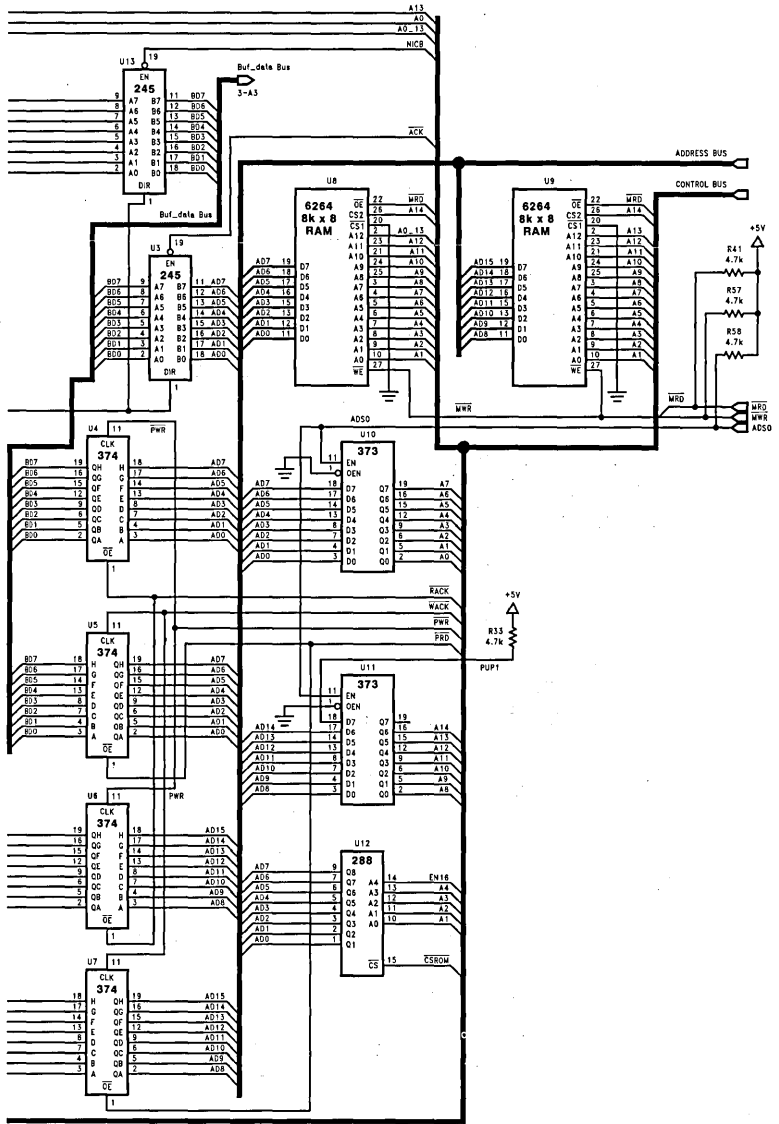
INT is just sent through the GAL to be buffered. The buffered signal which comes out of the GAL is INTO. The NCSROM is a very simple signal as it consists only of AD14 and NMRD. AD14 comes from the NIC and selects either the PROM (when low) or the on-board RAM (when high).

LERIC/NIC PC-AT Evaluation Board



TL/F/11706-11

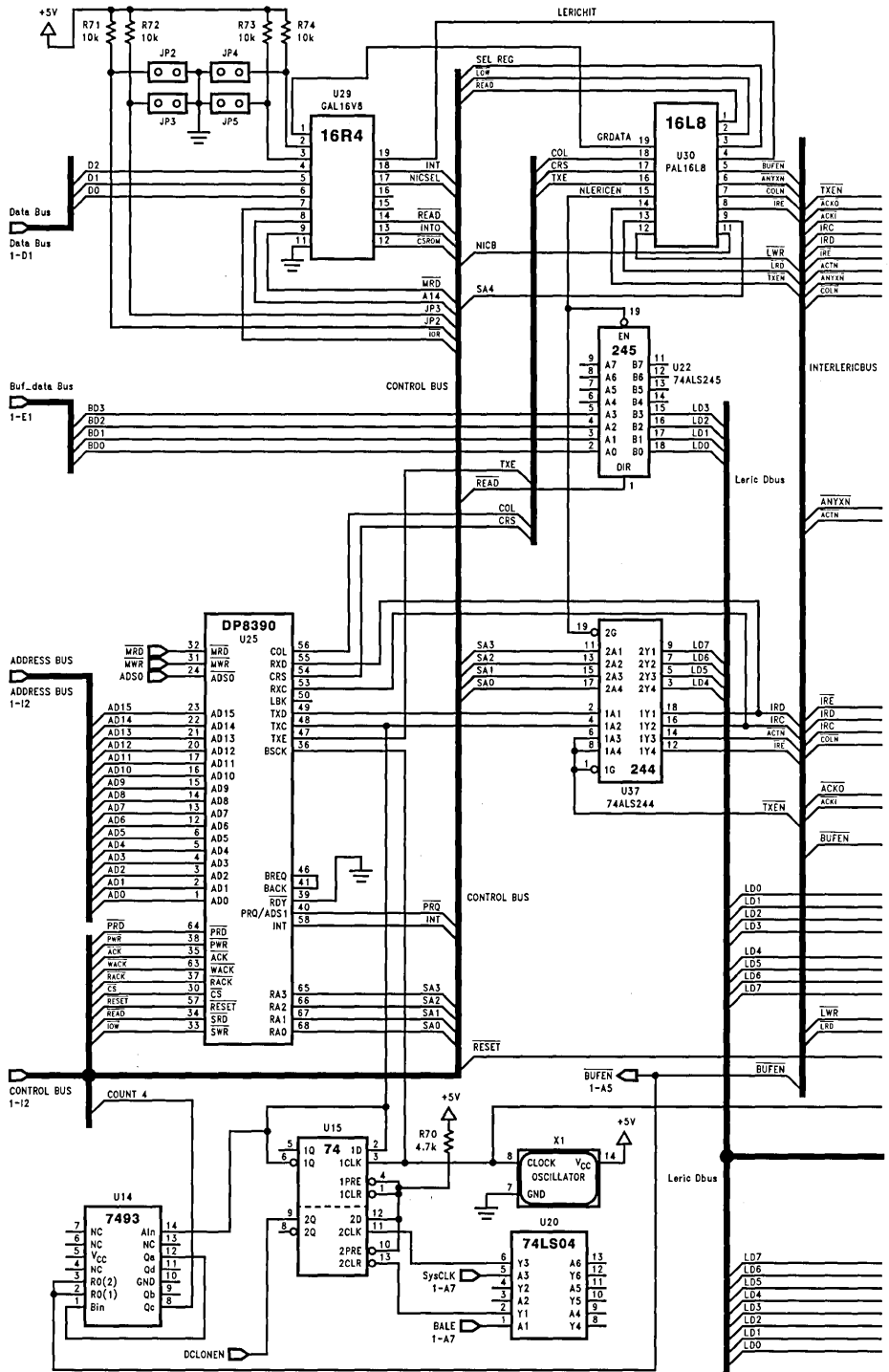
LERIC/NIC PC-AT Evaluation Board (Continued)



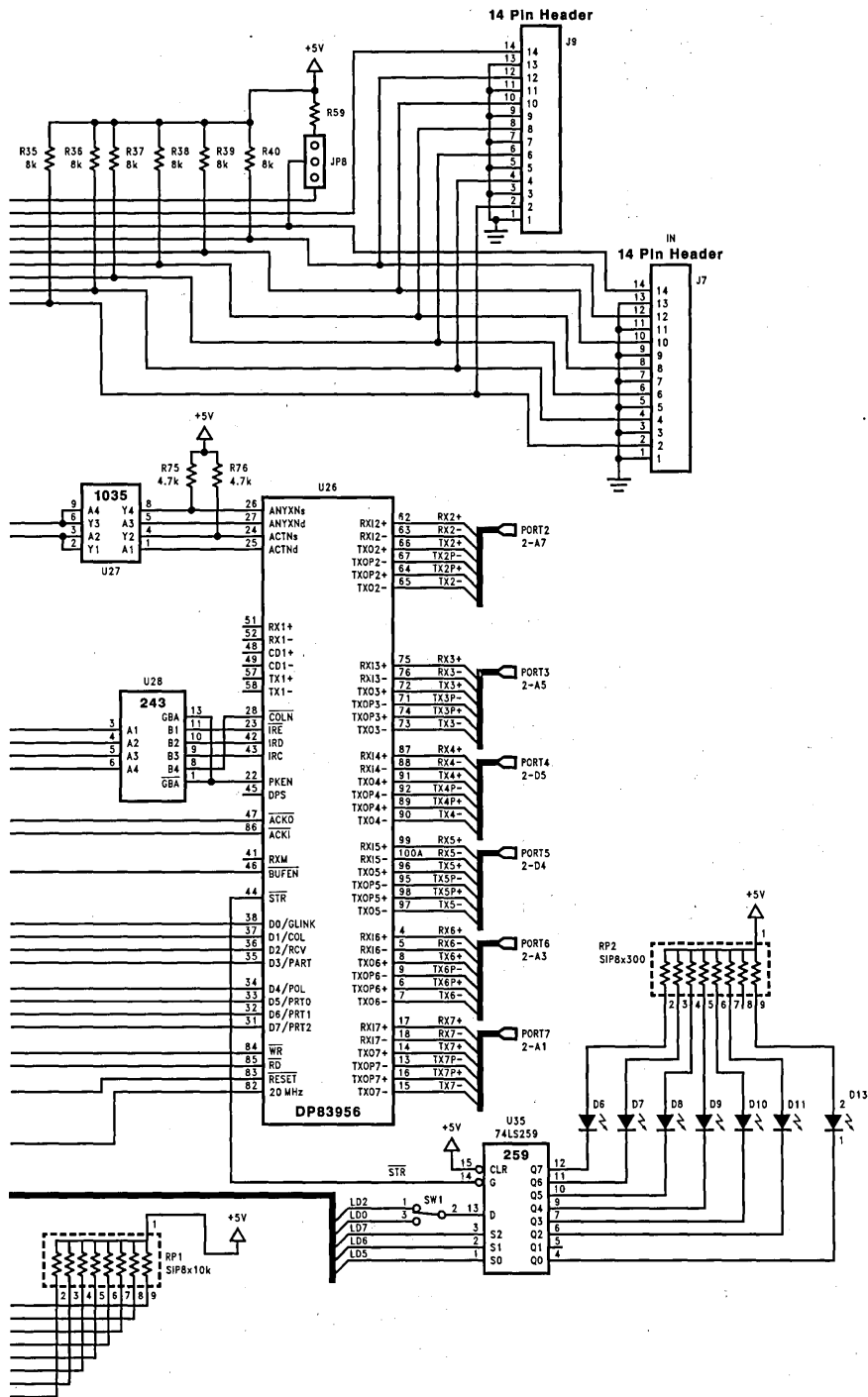
| AT Pins | Assigned Pins |
|---------|---------------|
| A2      | 02            |
| A3      | 03            |
| A4      | 04            |
| A5      | 05            |
| A6      | 06            |
| A7      | 07            |
| A8      | 08            |
| A9      | 09            |
| A10     | 10            |
| A11     | 11            |
| A12     | 12            |
| A13     | 13            |
| A14     | 14            |
| A15     | 15            |
| A16     | 16            |
| A17     | 17            |
| A18     | 18            |
| A19     | 19            |
| A20     | 20            |
| A21     | 21            |
| A22     | 22            |
| A23     | 23            |
| A24     | 24            |
| A25     | 25            |
| A26     | 26            |
| A28     | 28            |
| A27     | 27            |
| A29     | 29            |
| A30     | 30            |
| A31     | 31            |
| B1      | 32            |
| B3      | 34            |
| B4      | 35            |
| B9      | 40            |
| B10     | 41            |
| B13     | 44            |
| B14     | 45            |
| B23     | 54            |
| B24     | 55            |
| B25     | 56            |
| B29     | 60            |
| B31     | 62            |
| C11     | 11            |
| C12     | 12            |
| C13     | 13            |
| C14     | 14            |
| C15     | 15            |
| C16     | 16            |
| C17     | 17            |
| C18     | 18            |
| D2      | 20            |
| D16     | 34            |
| D18     | 36            |

TL/F/11706-12

LERIC/NIC PC-AT Evaluation Board (Continued)



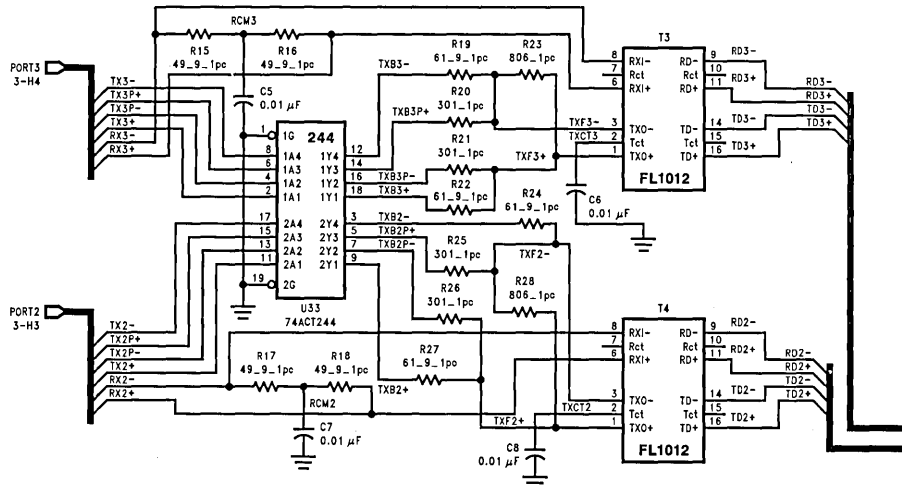
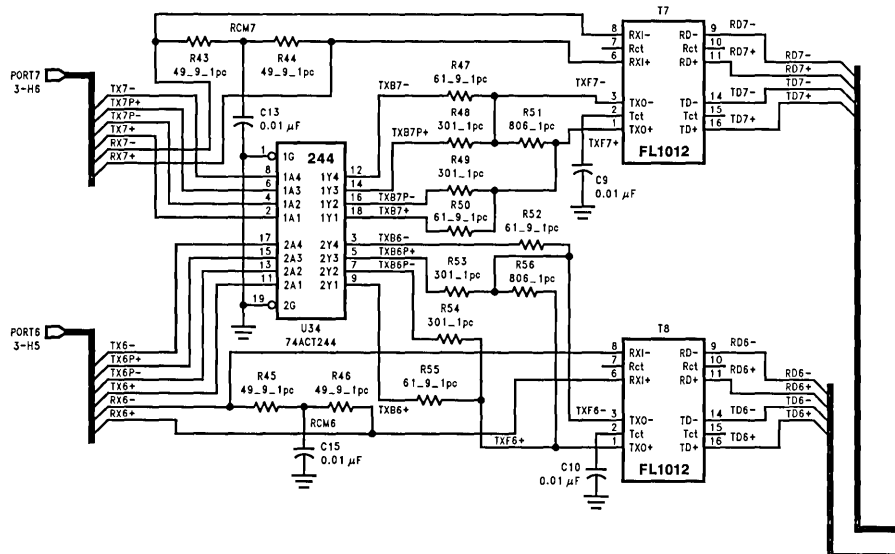
LERIC/NIC PC-AT Evaluation Board (Continued)



Note: All resistors to be 5%, 1/4W unless otherwise indicated.

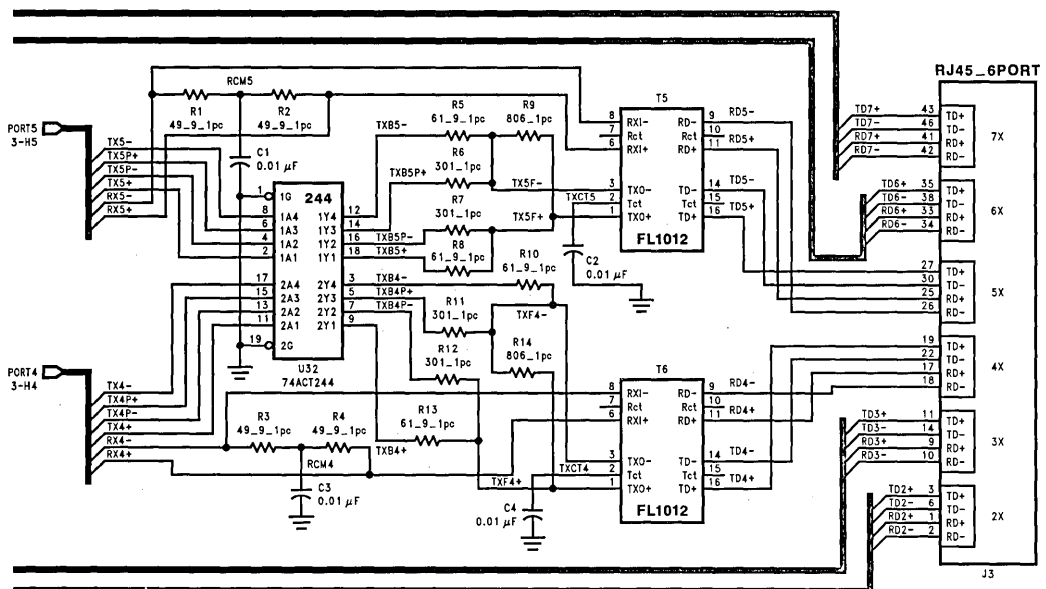
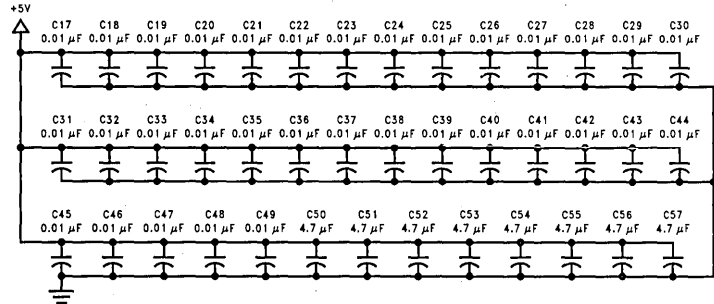


LERIC/NIC PC-AT Evaluation Board (Continued)



TLF/11706-15

LERIC/NIC PC-AT Evaluation Board (Continued)



TL/F/11706-16

Note: Resistor Nomenclature is as follows:

- 49\_9\_1pc = 49.9Ω, 1%
- 61\_9\_1pc = 61.9Ω, 1%
- 301\_1pc = 301Ω, 1%
- 806\_1pc = 806Ω, 1%

## 9.0 BILL OF MATERIALS

## Capacitors

|              |                                    |
|--------------|------------------------------------|
| 0.01 $\mu$ F | C1..C10, C13, C15, C17, C18....C49 |
| 4.7 $\mu$ F  | C50..C57                           |

## Miscellaneous

|                    |              |                                        |
|--------------------|--------------|----------------------------------------|
| *0.5mmGreenLED     | D6..D11, D13 |                                        |
| *SS-668806-NF      | J7           | (6PORT, 8 POSITION, RJ45 FROM STEWART) |
| *3__pin__jumper    | JP7, JP8     |                                        |
| *2__pin__jumper    | JP2..JP6     |                                        |
| *jumpers6x2        | JP4          | (jumper block)                         |
| *TOGGLESWITCH_DPST | SW1          |                                        |
| *FL1012            | T3..T8       | (VALOR)                                |
| *102160-2          | J7, J9       | AMP PIN HEADER                         |
| 746194-2           | J6, J8       | AMP RECEPTACLES                        |

## Resistors

|           |    |                 |                                                          |
|-----------|----|-----------------|----------------------------------------------------------|
| 8k        | 5% | $\frac{1}{4}$ W | R35..R40                                                 |
| 10k       | 5% | $\frac{1}{4}$ W | R29, R30, R42, R60, R71..R74                             |
| 49.9      | 1% | $\frac{1}{4}$ W | R1..R4, R15..R18, R43..R46                               |
| 4.7k      | 5% | $\frac{1}{4}$ W | R32, R33, R34, R41, R57, R58, R59, R70, R75, R76         |
| 61.9      | 1% | $\frac{1}{4}$ W | R5, R8, R10, R13, R19, R22, R24, R27, R47, R50, R52, R55 |
| 301       | 1% | $\frac{1}{4}$ W | R6, R7, R11, R12, R20, R21, R25, R26, R48, R49, R53, R54 |
| 806       | 1% | $\frac{1}{4}$ W | R9, R14, R23, R28, R51, R56                              |
| *SIP8x300 | 5% | $\frac{1}{4}$ W | RP2                                                      |
| *SIP8x10k | 5% | $\frac{1}{4}$ W | RP1                                                      |

## IC's

|                  |                |                          |
|------------------|----------------|--------------------------|
| PAL16L8 (15 ns)  | U16, U30       | PLCC Type (SOCKETED)     |
| GAL16V8 (10 ns)  | U1             | PLCC Type (SOCKETED)     |
| GAL16V8 (15 ns)  | U29            | PLCC Type (SOCKETED)     |
| PAL20L8D (10 ns) | U2             | PLCC Type (SOCKETED)     |
| 74ALS245         | U3, U22, U13   |                          |
| 74ALS374         | U4, U5, U6, U7 |                          |
| 6264RAM          | U8, U9         | 8k x 8 STATIC RAM 100 ns |
| 74ALS373         | U10, U11       |                          |
| *74S288          | U12            | SOCKETED                 |
| 74LS93           | U14            |                          |
| *27128           | U18            | EPROM (SOCKET ONLY)      |
| 74ALS244         | U19, U37       |                          |
| *20 MHz          | X1 (0.01%)     | Crystal Oscillator       |
| 74LS04           | U20            |                          |
| 74ALS243         | U28            |                          |
| 74ALS1035        | U27            |                          |
| DP8390           | U25            | NIC (SOCKETED PLCC)      |
| 74ALS74          | U26            |                          |
| 74ACT244         | U32, U33, U34  |                          |
| 74LS259          | U35            |                          |
| DP83956          | U36            | LERIC                    |

\*Note: Everything is surface mount except the devices that were marked with an "\*\*\*\*". devices with an "\*\*\*\*" are through hole.

# RFI Suppression Techniques in DP83950 (RIC) Based Systems

National Semiconductor  
Application Note 888  
James A. Mears



## CASE HISTORY—EXCESSIVE RFI FROM 12-PORT REPEATER

All commercial and consumer electronic equipment containing RF generating circuitry or devices must pass compliance tests for RF emissions before the equipment can be sold in the US (and most other countries). The equipment may not be legally marketed without first meeting the requirements of the FCC's rules (or those of the appropriate regulatory agency in the country where the equipment is to be marketed). Compliance testing for FCC requirements may be performed by the manufacturer or by one of many contractors specializing in this type of testing. Manufacturers who elect to self-verify must have the necessary equipment and an open-area test site (OATS) meeting the requirements of ANSI C63.4-1991 and FCC OST 55. In addition, complete verification test records must be maintained by the manufacturer for each device-type being produced. Failure to perform the required testing or marketing of non-compliant equipment can and often does result in severe legal penalties for the offending manufacturer.

Design for RFI suppression and compliance is frequently overlooked or given scant attention by equipment design engineers during the initial stages of product design. The result is often frantic and usually costly last-minute redesigns or modifications. In some cases, significant business and sales opportunities are lost or the product may never be successfully brought to market. It is therefore most important that RFI-proof design techniques be incorporated as an integral part of all engineering design specifications and procedures. These should begin at product concept and continue through to final sale and installation.

This application note is not intended as a comprehensive guide to all aspects of RFI-proof design. The techniques presented in this note resulted from on-site tests and equipment modifications in the example case only. National Semiconductor does not imply that if only the techniques discussed herein are incorporated in any design, that design will be rendered compliant. There are many other RFI-proof design approaches and methods that must be considered and may be found more effective in a particular situation.

### Description of the EUT and the Problem

The equipment under test (EUT) in the example system is an expandable, 12-port multipoint repeater. The device is designed to operate in a twisted-pair Ethernet environment. Functionality is under the control of National's DP83950 Repeater Interface Controller (RIC). The repeater has 12 twisted-pair ports served via RJ-45 connectors. The input port may be optionally fed via coax or fiber optic cable. The device is designed to be expandable with up to three other units.

Mechanical construction of the unit is conventional. All circuitry is contained on two cards mounted inside of a 2-piece steel enclosure. One large card having the majority of the circuit is permanently mounted. An externally removable plug-in module contains the input interface circuit. Components are primarily surface mounted with some through-

board mounted parts such as connectors. A power supply and fan are also mounted in the enclosure.

The inside of the enclosure is coated for increased conductivity. The pieces of the enclosure are joined with screws and all mating surfaces are bare of paint. The input module is inserted through an opening in the chassis and secured with screws. The chassis has other openings which allow mounting of the RJ-45 connector, indicator LED's, air inlet, fan exhaust and AC line power cable socket.

Excessive emission (RFI) relative to FCC Class A limits in the frequency ranges 30 MHz–120 MHz and 200 MHz–260 MHz was the primary problem with the system. Also, emission levels were considered marginal at other frequencies as shown in *Figure 1*. The task was to find the source(s) of the emissions and modify the device to reduce them to at least 6 dB below the specification limits. This was done by isolating the radiation mechanism and identifying the ultimate source of the energy. The process and what was found will be presented first. Next, the causes and cures of the RFI are detailed followed by recommended system design practices. In addition, a list of sources of relevant information is included in the appendix.

### Test Facilities

An OATS, *Figure 2*, and all necessary RFI testing equipment was available at the customer's facility. This speeded and eased diagnosis of the RFI problems. Corrections and modifications could be evaluated more readily than might have been possible with a remote or contract facility.

The OATS complied with the requirements of ANSI C63.4-1991 and FCC OST 55. Other facilities included a screen room, antennas and remotely controllable mast, turntable for the EUT, complete and automated instrumentation and capable, experienced EMC engineers to operate the equipment. Most diagnostic and all compliance tests were conducted on the OATS. Test repeatability on the OATS was found to be excellent. Some diagnostic and problem isolation testing were done in the screen room.

### TESTING FOR RFI

A number of different tests and techniques were used in the course of tracking down and isolating the RFI problems with this system. These fell into the following general areas of investigation:

- complete RFI scans (30 MHz–1000 MHz)
- shielding effectiveness of the case
- cable and connector shielding and radiation contribution
- internal shielding mechanisms of the case and PCBs
- contribution from peripheral circuits
- signal quality on the PCBs
- power supply and power supply bypassing
- test message traffic and number of ports operating

### RFI Scan

An RFI scan is a measurement method used to determine the level versus frequency of RFI emissions being produced

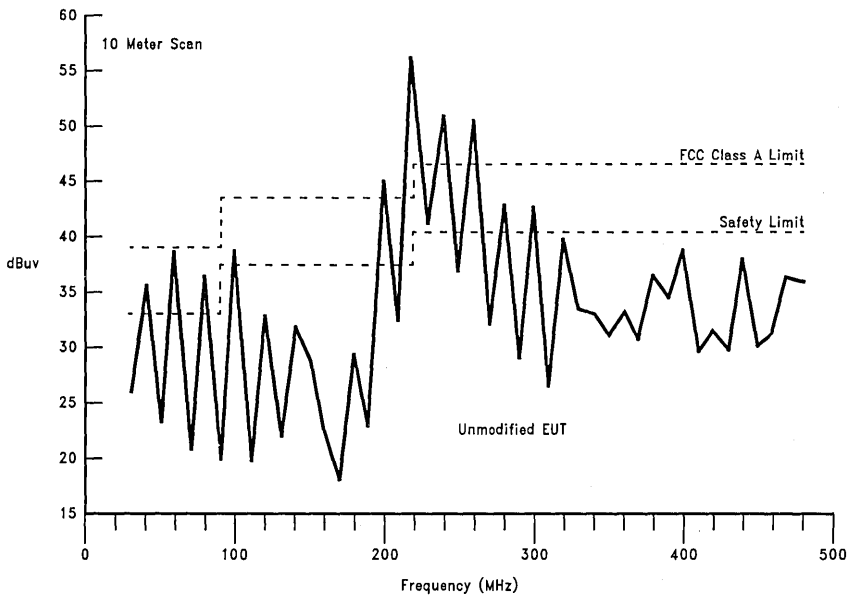


FIGURE 1. Unmodified EUT

TL/F/11821-1

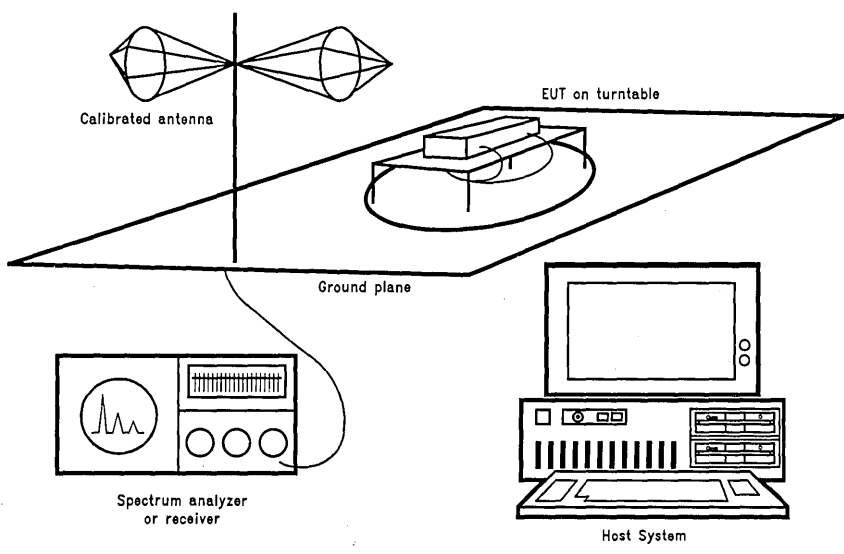


FIGURE 2. Equipment Configuration

TL/F/11821-2

by the EUT. In the test, an operating EUT is positioned on a rotating table 0.80 meters above a ground plane and 10 meters from a receiving antenna. A spectrum analyzer, or calibrated receiver designed for RFI testing, and a calibrated antenna are used to make the level measurements. The turntable azimuth, antenna height and polarization are changed to determine maximum emissions. The turntable is rotated through 360 degrees of azimuth while the receiver's output level is recorded for all frequencies within the specified range (30 MHz-1000 MHz). In combination with azimuthal rotation, the antenna's height is also varied from 1 to 4

meters to determine any elevational variations in emission level. In addition, scans are made with the antenna in both vertically and horizontally polarized modes when linearly-polarized antennas are used. Once a spectral signature is determined for the target device, certain frequencies at specific azimuths will predominate. These could be checked first after modifications are made to quickly assess the effectiveness of the change. This shortens test times appreciably. A complete scan should only be needed when sufficient improvements necessitate a new base-line reading.

RFI scans on the OATS were used to measure EUT RFI performance in the suspected trouble areas to be detailed. Scanning is a time-consuming procedure if done manually. Fortunately, this testing was partially automated in the case presented here.

#### **RFI Versus Port and Activity**

RFI qualification tests were eventually conducted with all ports active. But, first it was necessary to determine the contribution that each output made to the overall RFI level. Each port was tested, in turn, transmitting simulated data with an average density of message activity. As testing progressed, it became evident that the level of RFI emissions was a direct function of the port's distance from the RIC device. Later, this was an essential clue to tracking down the RFI generating mechanism.

#### **ENCLOSURE SHIELDING EFFECTIVENESS**

It was necessary to determine whether the device's case was an effective shield since openings and joints can leak radiation. The panels of the case can carry induced currents which show up as RFI. Also, the test would reveal if the cables were the radiation source instead.

To carry out the test, the paint was removed around all openings and joints for about 1 cm. All openings and joints were then covered, in turn, with self-adhesive copper foil. This attempted to isolate joints and openings as possible causes of RFI.

The greatest reduction was made when the 12-port, RJ-45 connector was covered over with copper tape (except for the port in use). Similar results were achieved when a "harmonica shield", designed for use with the RJ-45 connector, was substituted for the copper tape. After this modification, nothing else was found which greatly affected the radiation from the case. This pointed to the cables as the dominant radiators. But the ultimate RFI source was yet to be isolated.

Another test was carried out to further confirm the cables as the primary radiators. This was done by enclosing the entire unit in a Faraday shield. The unit was first inserted in a heavy, insulating plastic bag. The cables were brought out through small openings in the bag. The insulated unit was then wrapped in heavy aluminum foil with all seams double-folded. A 2.5 cm-wide braided, grounding strap was tightly folded in a seam in the foil along the length of the case. The ground strap was clamped to an earth-ground rod beneath the turntable on which the unit sat. A scan of the unit revealed almost identical radiation levels to the previous measurement without the shield. This further strengthened the view that the cables were the main external radiators.

#### **CABLE AND CONNECTOR SHIELDING**

Attention was now focused on cabling as the primary radiator. The contribution from each cable had to be determined and the main culprit identified. To do this, each cable was wrapped, in turn, in an aluminum foil/braided-copper shield along its entire length above the ground plane and grounded to the ground plane. The largest contributor by a substantial margin was the twisted pair followed by the coax and power cables, respectively.

It should be well known that the shielding effectiveness of twisted-pair line decreases with increasing frequency above a few megahertz. For signals above 10 MHz, it must be

considered as unshielded line. For this reason, any high frequency signals originating inside the enclosure that managed to couple onto the twisted-pair can easily couple into free space. However, substitution of shielded cables as a fix for this situation was out of the question.

When a shielded power cord was substituted for the normal one, radiation was increased. This larger antenna radiated more efficiently. It also pointed to the grounding wire as the pickup device. The contribution from the power cord was reduced by shortening the length of the safety ground (green wire) connected to the chassis from over 8 cm to about 3 cm.

The coax was found to be leaking some radiation, but the amount was small. It was felt that more could be achieved by concentrating on the twisted-pair cables as the mechanism. And, like the twisted-pair, substitution of another type of coax with a foil/braid shield was not possible. So, efforts were now turned to locating the source of the offending signals within the enclosure.

#### **Internal Case and PCB Shielding Mechanisms**

It was generally suspected that RFI produced by the operating logic devices on the PCB was being coupled out of the enclosure. The next job was to isolate and identify the contributors.

The PCB layout divided the board into two main areas: one contained the output driver circuits and filters; the other area contained the RIC, peripheral control, system oscillator and indicator circuits. A clear space across the PCB between these areas allowed a shield to be attached to the case top thus dividing the interior into two cavities. A scan with this arrangement produced lower radiation from the coax and slightly reduced radiation from the twisted-pair. Still, the overall unit was far from meeting FCC limits.

Two other experiments were tried at this time that did reduce the RFI but proved impractical from a manufacturing standpoint. In the first test, a grounded, copper foil shield was placed on the underside of the PCB insulated from the PCB by a thin plastic sheet. Called an "image plane", this reduced RFI and pointed to possible deficiencies in the PCB's internal ground plane or its connections to the case. It also indicated that transmission lines from the RIC to an output driver, unshielded by the PCB ground plane, were radiating.

In the second experiment, the size and location of grounding points between the PCB and the chassis was checked as a possible contributor. Larger-area connections were added from the PCB ground plane to the case. Radiation was reduced when the grounding connection at the RJ-45 connector was increased in size.

Results of these experiments pointed to signals associated with the RIC or its output circuits as possible causes of some of the RFI. Further investigations would concentrate on these areas. First, however, other circuitry would be checked for problems.

#### **Power Supply and Fan**

The power supply, an open-frame switching type, was checked next. The supply was disconnected from the circuit boards and a dummy load attached to its outputs. A scan revealed no significant RFI from the power supply alone.

The power supply was re-connected to the circuit board and the fan was disconnected. RFI in the frequency range 250 MHz–350 MHz was reduced. The fan was initially bypassed with only a 0.1  $\mu\text{F}$  ceramic capacitor. Evidently, this was not adequate. Addition of a 5  $\mu\text{F}/35\text{V}$  tantalum electrolytic capacitor on the PCB at the fan connector reduced radiation by an average of 3 dB over the above range.

### Logic Circuitry Power Bypassing

The PCB power distribution system was the next area of investigation. The number, location, type and size of bypass capacitors was examined. This check revealed that the number of capacitors was insufficient and would need to be increased. And, the capacitors would need to be relocated closer to the IC's for better effectiveness.

Noise across the power pins of the high-current consumption devices was checked. In several cases this noise approached a volt or more. In the original design, RF bypass capacitors (0.1  $\mu\text{F}$  ceramics) were placed about 1 to every 4 logic devices. Addition of capacitors across the power pins of the output drivers reduced supply noise by about half. This also reduced RFI in the lower frequency ranges.

Four RF bypass capacitors were located near and intended to serve the RIC device. It was evident from the switching noise in this portion of the PCB that bypassing would need improvement. Additional RF bypasses were added at the RIC's power pins as well as four 5  $\mu\text{F}/35\text{V}$  tantalum electrolytics arranged one per side. This improved the supply noise situation and also reduced the RFI below 100 MHz.

The bypassing at the PCB power supply connection point was also checked. The initial design used aluminum electrolytics paralleled with an RF ceramic. These appeared to be performing adequately and were not changed. See Appendix A for helpful bypass capacitor layout hints.

### Master Oscillator

All timing and data rate control signals were developed from one 40 MHz crystal oscillator device. RF bypassing at the oscillator appeared to be adequate and tests did not decisively pinpoint it as the cause of specific interference.

### Signal Quality on the PCB

Signal quality was the next area investigated. Signal aberrations like overshoot, crosstalk and ringing contribute appreciably to the RFI problem. Reducing or eliminating these problems correspondingly reduces RFI.

The twisted-pair port drivers were originally FACT devices. But, the twisted-pair Ethernet design does not specifically require either the high current drive or extremely fast rise times of which the FACT devices are capable. So, guided by RFI studies of several logic families made by Violette Associates for National's Digital Logic Division, HCT equivalents were substituted. This reduced RFI above 150 MHz by at least 3 dB, but more improvement was still needed. See Appendix B for additional information sources.

The transmission lines connecting the RIC to the output drivers had been previously identified for closer scrutiny. A look at the signals arriving at the unterminated port driver inputs revealed high levels of over/undershoot. Clearly, some type of termination would be needed to control the quality of these signals. There in, series and diode terminations were tried on the lines exhibiting the worst problem. Of

these, the series was the most effective at reducing overshoot. It had the additional advantage of being the easiest modification to incorporate on the prototype PCB.

Selecting the termination type led to the discovery that the RIC's output signal transition times were in the under 2 ns region. These signals were among the most active and longest signal paths in the system. A look with a spectrum analyzer identified troublesome frequencies as components of these signals. Perhaps here, together with the termination issue, was another root cause of the RFI problem. A small improvement here would likely produce a greater improvement in overall RFI.

Some experimentation showed that low-pass filtering of the RIC's output signals further reduced overshoot at the input of the output driver. Crosstalk with adjacent lines also was reduced. With only the longest transmission path thus filtered, the troublesome RFI frequencies were improved.

Following this test, all outputs were modified to add filtering, 50  $\Omega$  at the RIC output pin in series with the line and 30 pF from line input to ground. The overall result of modifications can be seen in the scan results plotted in *Figure 3*. When compared to the initial unmodified unit, a significant improvement is evident. The problem remaining was to improve the margin to the specification limit below 150 MHz; but, it was felt that this would require a new layout. The layout needed to incorporate the modifications found thus far together with improvements to power and ground planes, closer placement of the RIC to its output drivers, and grounding improvements. The fullest improvement would be evident only after all of these changes could be tested in concert.

### RFI SOURCES AND CAUSES

Now that the layers of the problem had been peeled away, several causes of the RFI problem could be identified. These were:

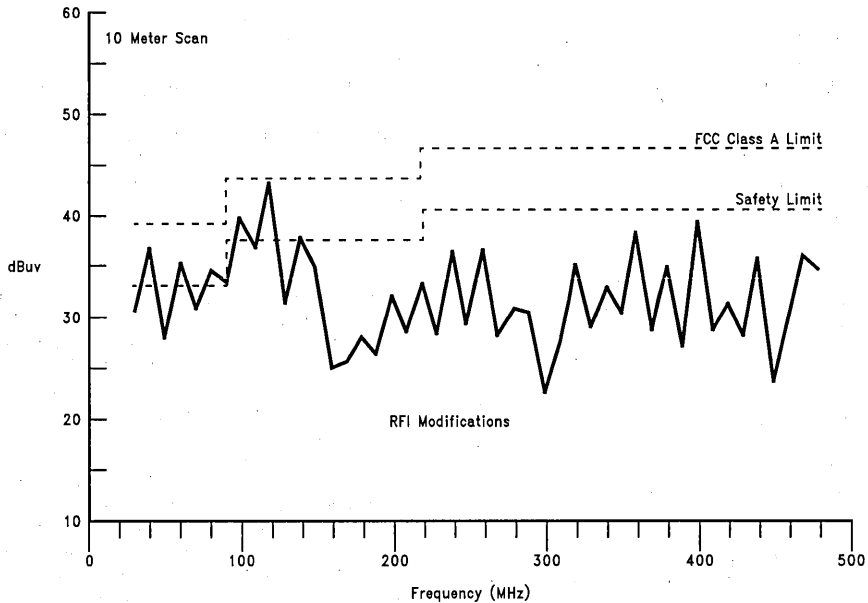
- excessive noise on RIC-to-driver signals
- excessive transmission line length
- insufficient bypassing with inefficient location
- compromised shielding effectiveness, and
- inefficient PCB layout

### Transmission Line Signal Quality and Excessive Length

The noise on the RIC-to-driver lines could be attributed to lack of adequate termination. The transmission lines were relatively high impedance, about 75  $\Omega$ . The lines also were long, over 10 cm, compared to the RIC output risetime which was in the sub-2 ns region. The lines were unterminated and lightly loaded by just the driver inputs. These conditions permitted excessive over/undershoot. Indeed, the lines over 8 cm in length exhibited 2V to 3V of overshoot and up to 5V of undershoot. Signal level was 12V peak-to-peak or more. Failure to control line length and provide termination contributed significantly to both RFI and crosstalk.

### Inadequate and Inefficient Bypassing

Excessive noise was found in the power supply system, as previously mentioned. Despite the use of power planes in



TL/F/11821-3

**FIGURE 3. EUT with RFI Modifications**

the PCB design, their impedance was excessively high. As such, the power system could not respond to the current demands of the logic devices. Also, the power system was not acting as a good image conductor for the transmission line system.

Several things were done to correct this situation. First, more RF bypass capacitors were added, as previously mentioned. In some cases, multiple capacitors with values of  $0.1 \mu\text{F}$  and  $0.01 \mu\text{F}$  were connected in parallel across the offending device's power pins. This was necessary to adequately control impedance over the operating frequency ranges of the device. Second, LF bypassing in the form of  $5 \mu\text{F}$  tantalum capacitors was added, one to every two high current drivers. Several more were placed at other locations on the PCB, particularly near the RIC. This was done to control low frequency noise and reduce lower frequency RFI emissions.

With these changes mandated, a re-layout was clearly called for. At that time two other problems with bypassing would be corrected. These were the lengths of conductors connecting IC power pins to the planes and the placement of bypass capacitors. Conductor length would have to be shortened and capacitors moved closer to the device requiring the bypass.

#### **Shielding Effectiveness**

Experiments indicated the need to improve shielding both on the PCB and in the enclosure. More isolation was need-

ed between the RIC and its peripheral circuits. This could be done with a shield in the case as previously mentioned. Shielding for the RJ-45 connector would be needed together with improved grounding to the case along its length. Ground plane contact area to the case would be increased. And ground plane coverage under transmission lines on the PCB would be extended to twice the minimum line-to-plane spacing for better coverage. All of this was in addition to correcting overall signal quality and bypassing.

#### **Layout Problems**

Since another layout would be done, several other things contributing to the RFI problem could be corrected. Grouping of the circuits would be improved. In particular, the distance from the RIC to its farthest output port drivers was as much as 25 cm. The objective would be to reduce these transmission line lengths by half. Peripheral circuitry on the PCB might need to be moved to do this. However, the power supply, connectors, mounting points and similar items could not be relocated for manufacturing reasons.

#### **Other Problem Areas**

Strong 30 MHz, 50 MHz, 70 MHz and 90 MHz signals from the area of the RIC, 40 MHz from the oscillator and components from the fan were noted as potential problems, but it was thought that changes to bypassing, layout and shielding would correct these.



### RFI Radiation Mechanisms

Several mechanisms were finally identified by which RFI was being radiated. The primary mechanism was radiation and crosstalk from the RIC-to-driver lines and thereby to the output twisted pairs. A secondary mechanism was through the power supply system due to inadequate bypassing. This was allowing excess noise on the grounding system for all signals. The third component was through reduced shielding provided by the internal power and ground layers of the PCB. This should have provided suppression of radiation and interaction of signals on the PCB. Other mechanisms included inadequate shielding and isolation between sensitive parts of the system and noise sources, fan bypassing and direct radiation from the RJ-45 port.

### CORRECTIVE ACTION SUMMARY

The encouraging results from the modified system made it practical to proceed with a full revision of the unit. It was anticipated with a high degree of confidence that the result would be a production-worthy and fully FCC-compliant system. In summary, the changes made to the unit were:

- changed peripheral logic from FAST and FACT to LS, ALS and HCT
- added series termination and filtering to RIC outputs
- revised and improved power/ground plane layout and coverage

- improved PCB grounding to case
- improved layout of differential lines from RIC to drivers (See Appendix A for details)
- tightened-up layout between RIC, output drivers and RJ-45 connector
- added shielding to RJ-45 connector
- added and improved RF bypassing for high-current-demand IC's and RIC
- added tantalum bypass capacitors (LF bypassing)
- improved fan bypassing and
- improved placement of peripheral circuits and indicators.

### RESULTING PERFORMANCE IMPROVEMENTS

The performance improvements in the production unit as the result of the above revisions can be seen in the new scan, see *Figure 4*. These are the corresponding measurements under FCC Class A test conditions to those in *Figures 1 and 3*. (For ease of comparison, all are plotted in *Figure 5*.) The EUT has been brought into compliance and with a healthy safety margin. It should be emphasized that these tests were carried out with all 12 ports operating and with the same traffic and messages. Later tests of a multi-unit system yielded results similar to the single unit system. All variations tested thus far have been fully compliant.

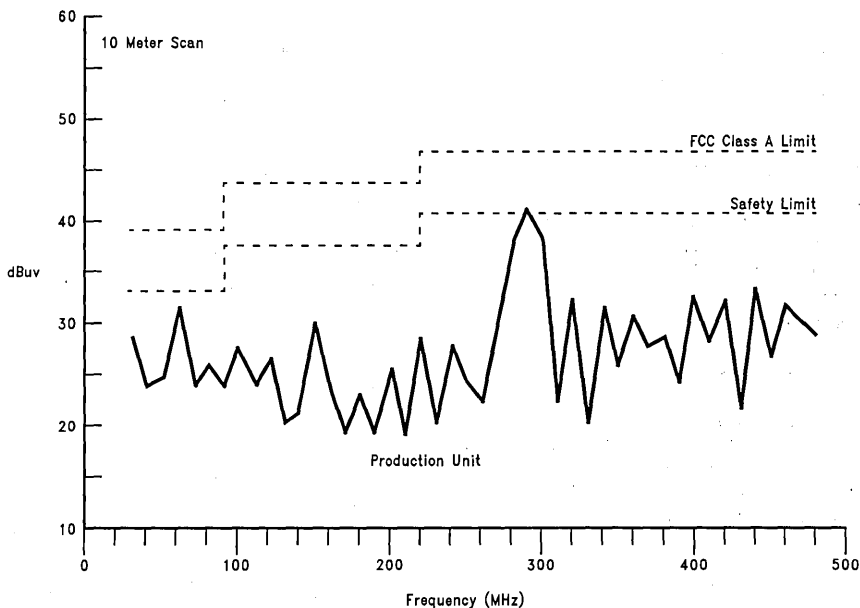
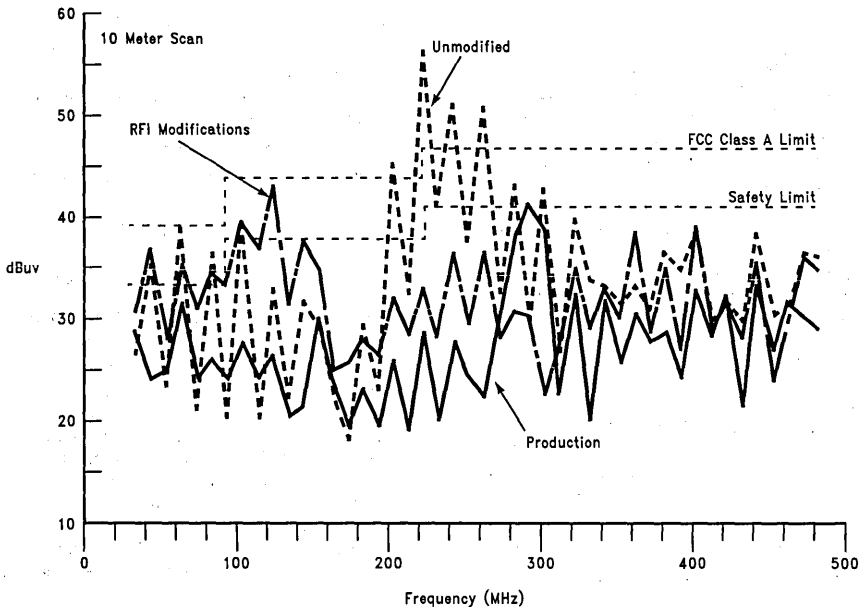


FIGURE 4. Production Unit

TL/F/11821-4



TL/F/11821-5

FIGURE 5. Comparison All EUT Types

From these tests and the layout, it is clear that RFI is a design issue that cannot be ignored until after the product is ready for market. It must be an integral part of the product's specifications and design from the beginning. Failure to do so can be extremely costly.

#### RECOMMENDED SYSTEM DESIGN PRACTICES

System designs using the RIC can benefit from careful attention to the design practices discussed in the topics which follow. Such practices can greatly reduce problems associated with RFI testing and qualification. Their adoption as a part of existing design standards is highly recommended.

#### System Design Hierarchy

Design for RFI compliance must be high on the list of system design requirements. This is especially true for devices which broadcast pulse signals over wire. Meeting regulatory requirements is made easier if a systematic approach is used in the design process. It is also a fact that systems designed for minimum EMI/RFI are more resistant to ESD and are subject to fewer signal-related problems.

The main elements of a good system design approach are, in order of importance:

- system specification including regulatory requirements and RFI
- signal quality standards and RFI-proof design practices
- testing methodology and requirements
- manufacturability considerations

- power system, supply and bypassing requirements
- transmission line system and terminations
- system mechanical, thermal and environmental requirements
- logic system design and functionality (initially, logic-technology independent)
- choice of appropriate logic technologies and other components
- layout and organization of PCBs, enclosures, cabling, etc.
- prototype evaluation and rigorous testing, and compliance testing.

#### Bypassing

The importance of good power system bypassing cannot be over-emphasized. Bypassing is the key ingredient allowing maximum system and component performance. The correct choice and application of bypass capacitors should be based on measured electrical performance and not on "rules-of-thumb" or unsubstantiated recommendation.

Bypass capacitors should be characterized for attenuation versus frequency. All capacitors are not created equal. Moreover, one size cannot necessarily perform best in all situations. The correct combination of capacitors is one which achieves adequate suppression of RFI-contributing power system noise.

In RIC-based designs, the following bypassing is recommended:

- Locate bypass components close to the RIC's  $V_{CC}/GND$  pins.
- Use no less than two, 0.1  $\mu F$  ceramic caps on each side of the RIC.
- Use one, 5  $\mu F$  to 10  $\mu F$  tantalum capacitor per side.
- Use one, 0.1  $\mu F$  per O/P driver; one, 5  $\mu F$  to 10  $\mu F$  per 2 O/P drivers.
- Use one, 0.1  $\mu F$  per octal driver; one, 5  $\mu F$  to 10  $\mu F$  per 2 octal drivers.
- Use one, 0.1  $\mu F$  per 2 SSI logic devices; one for each synchronous device.
- Use one, 5  $\mu F$  to 10  $\mu F$  per 4 SSI logic devices; one for every 2 synchronous devices.
- At PCB power entrance points use a 0.1  $\mu F$  and a 10  $\mu F$  per supply voltage.
- For DC fans (if used) use a 0.1  $\mu F$  and a 10  $\mu F$ .
- Use a Pi-filter (or longitudinal choke) for oscillator  $V_{CC}$  power.

**Note:** All ceramic capacitors are RF-rated types, leadless-monolithic preferred. Electrolytic capacitors are solid-electrolyte, tantalum types. Tantalum capacitor voltage rating should be a minimum of 5X the power supply voltage.

### Layout Recommendations

A disorganized component layout can contribute to both signal and RFI problems. When laying-out a RIC-based design, observe these precautions and recommendations.

- Use a multi-layer PCB with dedicated power/ground planes.
- Keep layout compact with RIC close to output drivers.
- Locate less critical peripheral and indicator circuits farther away.
- Locate output connector and filters close to RIC output drivers.
- Layout to minimize transmission line lengths from RIC to drivers.
- Provide frequent and generously sized grounding pads for case ground points.
- Design in extra locations for bypasses. Omit the capacitors if tests show them to be unnecessary.

**Note:** It is easier to remove unnecessary components from a PCB than it is to add needed ones after the board is built. This is especially true for surface-mount PCB's.

### Transmission Lines

An efficient layout also must consider the transmission lines. Particular attention should be paid to the following recommendations:

- Keep lines short and direct.
- Extend ground plane under all transmission lines.
- Use fully shielded lines (stripline) for high-level signals.
- Terminate all lines exhibiting over/undershoot or crosstalk noise.
- Observe pairing of differential lines from RIC outputs (*Figure A3*) (Appendix A).
- Maintain at least twice the transmission line's width between pairs of differential lines.
- Terminate RIC O/P's to reduce reflections, overshoot and noise. Series terminations with a value of  $Z_0 - 10\Omega$  are recommended.
- LP filter RIC outputs, if necessary, to reduce noise associated with fast output transitions. The capacitor value should be chosen for a 5 ns time constant in conjunction with the series termination resistor's value.

### Recommended Logic Device Types

In any logic system design it is wise not to employ devices with performance characteristics exceeding those required to adequately handle the system's frequencies or signals. Higher performance devices (usually taken to mean frequency handling and rise times) normally produce increased amounts of RFI over a broad spectrum. To save RFI difficulties, do not put in more performance than the design needs. The following device types have been tested and found to work well and reduce RFI in RIC-based designs:

- HC or HCT for differential line driver circuits
- LS, ALS or HC for peripheral circuits, interfaces and LED drivers.

### Oscillator Recommendations

Though often overlooked, the choice and use of oscillator components can greatly affect system RFI performance. The following are the recommended design practices for RIC-based systems. (These apply equally well to any logic system).

- Metal can, grounded-case oscillator modules are preferred. In general, plastic-case types have inadequate shielding and are not recommended.
- Supply oscillator power through a Pi-section filter or longitudinal choke.
- Observe proper supply bypassing.
- Locate oscillator close to the RIC.
- Keep transmission lines short and well shielded.

## APPENDIX A—CIRCUIT AND LAYOUT DETAILS

## Bypass Layout

Poor layout will seriously handicap even the best bypass components. Bypass components must be placed in close proximity to the point where impedance control is needed. Any excess inductance between the capacitor and the signal source (usually an IC) increases the effective impedance of the network. This decreases the effectiveness of the by-

passing. Bypassing is often called impedance compensation. The extremely fast energy demand impulses produced by high-speed IC's, especially CMOS, require an equally fast response from the power system supplying them.

Figure A1 shows how to locate bypass capacitors for good performance in an SOIC layout. Figure A2 shows the layout for PCC device packaging. Of course, differences in the power/ground pin organization of the device may necessitate a different placement of bypass components.

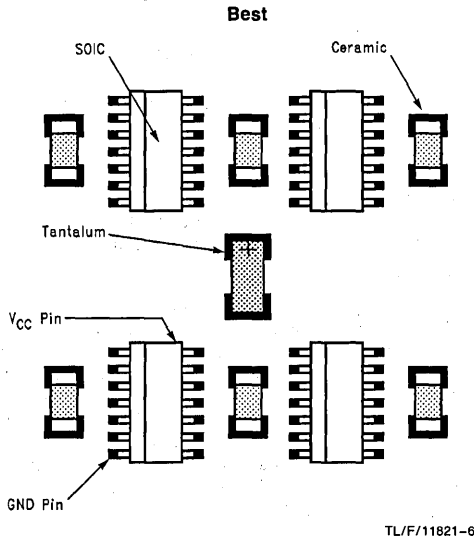
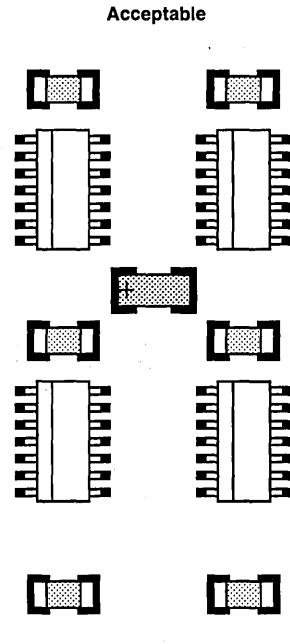


FIGURE A1. SOIC Bypass Capacitor Placement



TL/F/11821-7

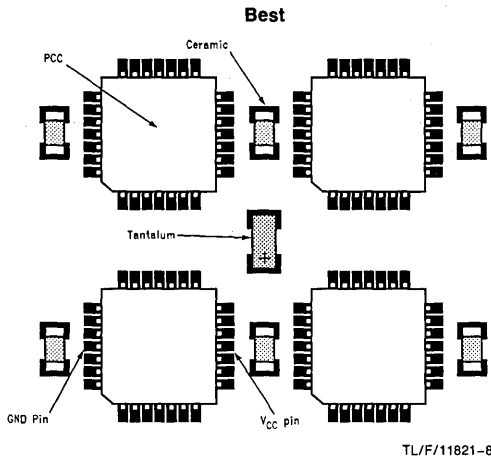
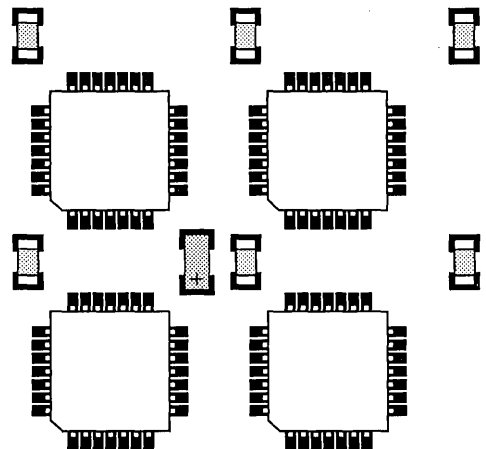


FIGURE A2. PCC Bypass Capacitor Placement (Typical)



TL/F/11821-9

### Differential Line Layout

Differential transmission lines require additional care in layout if they are to function correctly. *Figure A3* illustrates both correct and incorrect ways of differential line layout. The spacing of differential lines affects their even or odd-mode characteristic impedance. It also affects coupling to adjacent lines. Since crossfalk is a function of line spacing, a good rule to observe is to allow at least twice the spacing of the differential pair between pairs.

Maintain equal lengths for both conductors by avoiding crossover and layer-change situations. If a crossover or direction change is made in routing the lines, then an opposite change should be made elsewhere in the lines to compensate the resulting length difference. Mitering corners also aids in preserving signal quality and impedance uniformity.

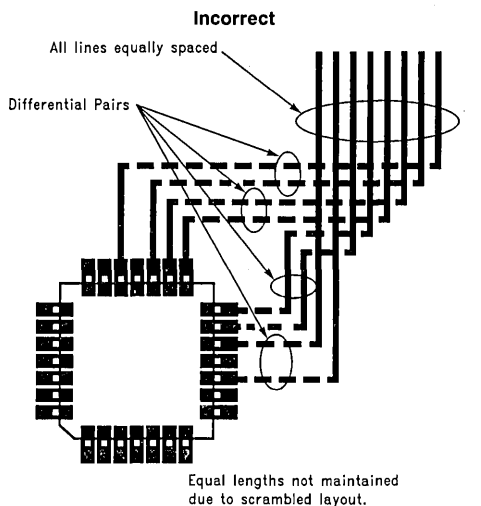
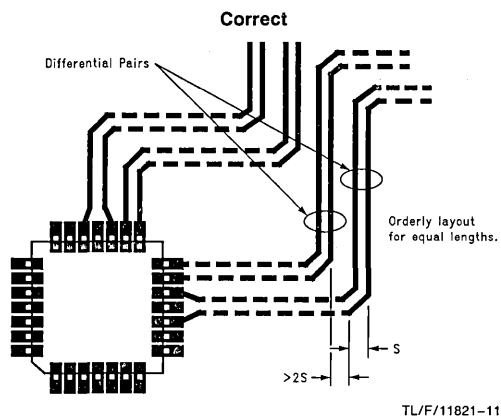


FIGURE A3. Differential Line Layout



### $\pi$ -Filter

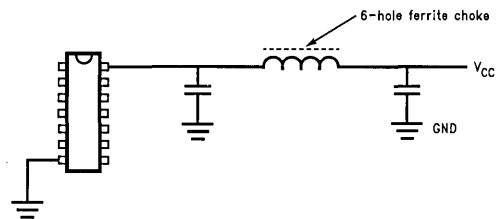
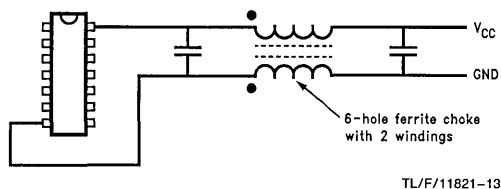


FIGURE A4. Oscillator Isolation Techniques

### Oscillator Supply Isolation

Oscillators and other frequency generating devices operating above a few megahertz should be isolated from the power supply system. This is done to prevent their becoming the dominant interference signal both on the PCB as well as in free space. Two convenient methods are shown in *Figure A4*. Both have the same component count; the only difference is the way in which the inductor is used. The pi-filter uses a simple ferrite-loaded inductor as part of a broadband filter. The longitudinal choke uses a ferrite-loaded transformer as a bucking choke. It is, in effect, a form of pi-filter in which the effects of opposing AC currents are made to cancel. Ferrite inductors like those illustrated are available from several sources: Siemens, Fair-Rite, Ferroxcube and Arnold.

### Longitudinal Choke



**APPENDIX B—ADDITIONAL INFORMATION**

National's application note library contains more information pertaining to the design of high-performance and RFI-proof systems. Some of these are listed below.

**Application Notes**

AN817—"Taking Advantage of ECL Min-Skew Clock Drivers"

AN467—"Surface Mount: From Design to Delivery"

AN393—"Transmission-Line Effects Influence High-Speed CMOS"

AN389—"Follow PC-Board Design Guidelines for Lowest CMOS EMI Radiation"

**Databooks**

400028—F100K ECL Logic Databook and Design Guide

**Bibliography**

"Antennas"; J.D. Kraus, Ph.D.; McGraw-Hill; 1950 (THE seminal work on antennas and radiation)

"Communication Systems: An Introduction to Signals and Noise in Electrical Communications"; A. Bruce Carlson; McGraw-Hill; 1968

"Grounding and Shielding Techniques in Instrumentation, 2nd Ed."; Ralph Morrison; John Wiley & Sons; 1977

"Code of Federal Regulations 47 (CFR 47) FCC Part 15—Radio Frequency Devices"

CFR 47 Part 2—"Frequency Allocations and Treaty Matters: General Rules and Regulations; Sub-part I, Marketing of Radio Frequency Devices; sub-part J, Equipment Authorization Procedure"

89-336 EEC—"EMC Directive of the European Economic Community"

EN55022 (CISPR 22)—"Radiated and Conducted Emission Limits (CENELEC)"

FCC OST 55—"Characteristics of open-field test sites (Aug. 1982)"

ANSI C63.4-1991—"Methods of measurement of radio noise emissions from low voltage electrical and electronic equipment in the range of 9 kHz to 40 GHz."

# DP83956EB-SA Stand Alone Hub

National Semiconductor  
Application Note 896  
Edward Boyd



## 1.0 INTRODUCTION

The DP83956EB-SA board is a low cost, stand alone, local area network repeater hub with minimal status display. The board is designed to demonstrate National Semiconductor's LiTe Repeater Interface Controller (LERIC™—DP83956). This stand alone hub solution contains 12 twisted pair ports, an Attachment Unit Interface (AUI) port, and a coaxial transceiver interface (CTI) port. The board allows for cascading with another DP83956EB-SA or a DP83956EB-AT board (6 port repeater hub card for use in an AT-Bus). The board requires a single +12V supply to become operational. The DP83956EB-SA board uses surface mount components to minimize size. Below is a list of the major features of the DP83956EB-SA board:

- 14 ports: 12 twisted pair ports, 1 CTI port, and a AUI port
- Powered by single +12V DC supply
- Jumper for easy termination of coax port
- 4 LEDs for real-time hub status
- Surface mount components for small size
- Designed for minimal EMI radiation

- Cascading headers for daisy chaining with another DP83956EB-SA or a DP83956EB-AT board
- Designed for low cost
- Stand alone
- Expansion header for full LED status, LERIC register access, or hub status counters
- Easy to operate with a single switch for LERIC mode load

## 2.0 USER'S GUIDE

### 2.1 Start Up

The DP83956EB-SA board should be connected to a 12V DC supply at right-angle header J1. The polarity of the input power should match 12V and GND markings next to J1. For proper operation of the coax port the jumper shunt on JP19 should be removed. The RESET button, S1 forces the LERIC controllers to restart, which will reset the consecutive collision counters, unpartition all ports, and perform a mode load with the options on switch S1. When the DP83956EB-SA is powered up or the RESET button is pushed, all four status LEDs will light up for one second.

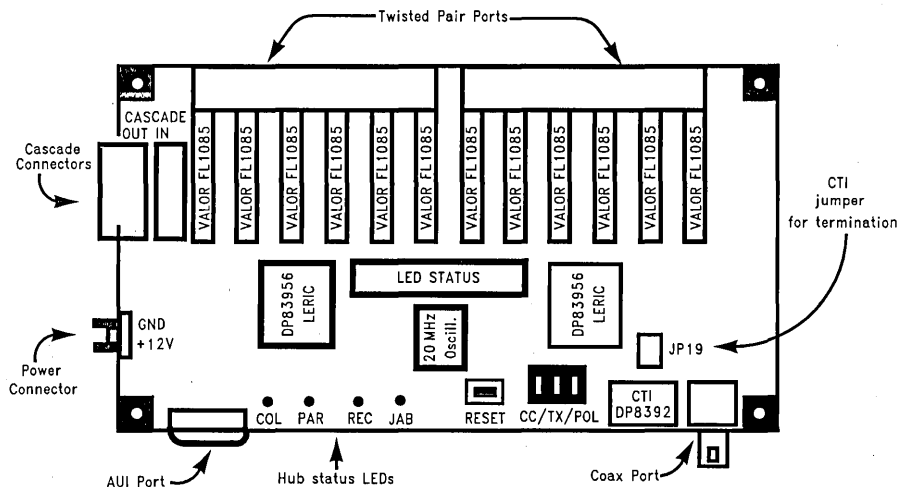


FIGURE 1. DP83956EB-SA Board

TL/F/11849-1

**2.2 Mode Load Options**

Three of the LERIC mode load options are available to the user on switch S2. This switch configures both of the LERICs upon reset or power up. S2 is labelled CC/TX/POL corresponding to the options in that order. The CC switch sets the consecutive collision limit for the LERICs. The LERICs will partition a port after 31 or 63 consecutive collisions on that port, depending on the CC switch value. The CC switch allows the user to set the consecutive collision limit for all fourteen ports on the board. When in the OFF position, the LERICs will partition after 31 consecutive collisions. The ON position will configure the LERICs for 63 consecutive collisions.

The switch labelled TX configures the unpartition on transmit only bit (TXONLY). If this bit is selected the LERICs will only unpartition a port after a good packet (532 bits or more without a collision) has been transmitted on that port. Without this bit set, the LERICs will unpartition a port after a good packet is transmitted or received. When the switch is in the OFF position, unpartition on transmit only is not selected, while in the ON position, TXONLY is selected on all fourteen ports. Switch number 3, POL enables or disables the polarity reversal option on the twisted pair ports (ports 2-13). This switch has no effect on ports 1 and 14. When the POL switch in the OFF position, the twisted pair ports will switch the polarity of the receive lines when inverted packets or link pulses are received on that port. If this switch is in the ON position, only packets and link pulses with the correct polarity will be recognized by the LERICs. More information on these functions is contained in the LERIC data sheet under mode load. Below is a table of the MLOAD options.

**TABLE I. MLOAD Quick Reference Guide**

| SWITCH | LERIC Data Sheet Name | ON Position                                    | OFF Position                                               |
|--------|-----------------------|------------------------------------------------|------------------------------------------------------------|
| CC     | CCCLIM                | 63 Consecutive Collisions to partition a port. | 31 Consecutive Collisions to partition a port.             |
| TX     | TXONLY                | Ports unpartition on good transmissions only.  | Ports can unpartition on good receptions or transmissions. |
| POL    | EPOLSW                | Polarity reversal on ports 2-13 is disabled.   | Polarity reversal on ports 2-13 is enabled.                |

Note: Options are loaded only during power up or reset.

**2.3 CTI Termination Jumper (JP19)**

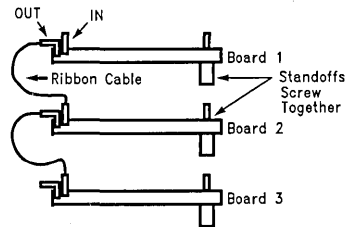
Included in the coaxial transceiver logic is a jumper (JP19) and a shunt for terminating the coax transceiver. When not

connected to a coax cable with proper termination, the CTI will continuously detect collision and the LERIC will partition the port. To avoid receiving these false collisions and prevent the unnecessary partitioning of the port, JP19 was included to allow for termination of the CTI when it is not connected to a network segment. Placing a shunt on JP19 will terminate the CTI. When the coax port is going to be connected to a properly terminated network segment, the jumper shunt should be removed.

**2.4 Cascading Boards**

Cascading the DP83956EB-SA board with more DP83956EB-SA boards is as simple as connecting the ribbon cable and standoffs. No additional configuration is required. A short ribbon cable should be connected to the connector marked "CASCADE OUT" with the other end connected to the "CASCADE IN" of the board below in the stack. (See Figure 2.)

When cascaded, the repeater functions as a single hub. A packet received on any of the cascaded boards will be transmitted on all boards at the same time, so cascading boards via the interLERIC bus adds no additional delay to the network. The hub status LEDs will only monitor the status of activity on its board's ports with the exception of the jabber LED. The jabber LED will light up on all cascaded boards when jabber is detected, regardless of the port receiving the packet.



**FIGURE 2. Cascading Structure**

TL/F/11849-2

**2.5 LED Display**

The LED display shows hub status for the board. If any port is receiving on the board, the green REC LED will light up. If any port on either chip is experiencing a transmit or receive collision the COL LED will be on. The APART LED will be on when any of the 14 ports on the DP83956EB-SA is partitioned. If the CTI is not properly terminated and the termination jumper (JP19) is off, the CTI port will be partitioned and the APART will go on. The jabber (JAB) LED will go on when a packet of excessive length has been received by any of the 14 ports. When boards are cascaded, the jabber LED will be lit on all boards when a jabber packet is received. Below is a quick reference chart for the LED status display.



TABLE II. LED Display Quick Reference

| LED                   | OFF                                                        | ON                                                     |
|-----------------------|------------------------------------------------------------|--------------------------------------------------------|
| Any Partition (APART) | None of the ports on the board are partitioned.            | One or more ports are partitioned on the board.        |
| Any Collision (ACOL)  | None of the ports on the board are experiencing collision. | A collision is occurring on one or more of the ports.  |
| Any Receive (AREC)    | None of the ports are receiving activity.                  | One or more ports are receiving activity on the board. |
| Jabber (JAB)          | A jabber packet is not being received or transmitted.      | A jabber packet is being received or transmitted.      |

### 3.0 A REVIEW OF THE DESIGN

The schematic for the DP83956EB-SA board is divided into functional blocks that appear on the different pages. The first page shows the interconnection of the functional blocks that make up the board along with the +12V power connector, power regulator for +5V output, and the decoupling capacitors. *Figure 4* shows a simplified block diagram of the DP83956EB-SA board. A daisy chain structure is required for the acknowledge in ( $\overline{\text{ACKI}}$ ) and the acknowledge out ( $\overline{\text{ACKO}}$ ) signals, while the remaining inter-LERIC bus signals are in a parallel bus configuration. The interacknowledge ( $\overline{\text{INTACK}}$ ) signal connects the LERICs on a single board.

### 3.1 Twisted Pair Ports 2-13

The logic to construct a twisted pair port begins at the LERICs (Schematic sheet 2-LERIC connections). Each port contains 6 pins (+TX, -TX, +TXOP, -TXOP, +RX, -RX). The four transmit pins  $\pm\text{TX}$  and  $\pm\text{TXOP}$  are filtered by a simple RC low pass filter to reduce noise (Schematic sheet 9). The four transmit lines then enter a simple TTL line driver (74HCT245) which provides the power to drive the twisted pair line (Schematic sheet 10). The transmit signals are then connected to Pulse Engineering's PE-65438 (Valor FL1085) which provides the magnetic materials for shaping the waves and an isolation transformer. The PE-65438 contains wave shaping resistors selected for use with a 74HCT driver to reduce signal jitter, provide a 100 $\Omega$  input impedance, and provide the proper output signal. The 245 was used because its pinout allowed for a cleaner layout (See layout sheet 1). *Figure 4* shows the path for the twisted pair ports and the items contained in the PE-65438. The receive path uses a common mode to reduce noise, a low pass filter to remove high frequency noise, and a 1:1 transformer for isolation. After the PE-65438, the twisted pair port is connected to a shielded modular jack. The shield of the modular jack is connected to chassis ground, which connects to the mounting holes for the board.

### 3.2 AUI Port 14

The connections for a full AUI port are shown on *Figure 11* of the schematic. From the LERIC's AUI port to the D connector only a 1:1 isolation transformer and some pull down resistors are required to construct the AUI port. The differential pairs are the same length and run in parallel as can be seen on *Figure 5*. To make the full AUI port of the LERICs into a twisted pair port, a twisted pair transceiver such as National's DP83922 would be required along the support logic. See the DP83922 data sheet for more information.

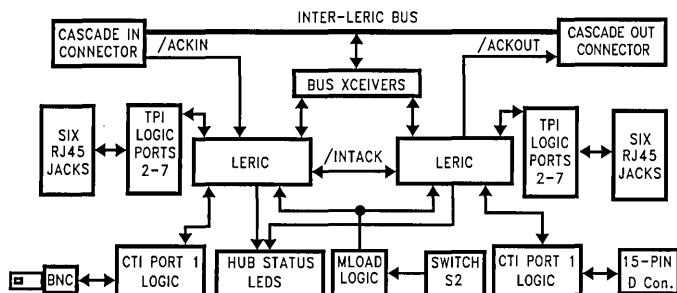


FIGURE 3. Block Diagram of the DP83956EB-SA

TL/F/11849-3

### 3.3 CTI Port 1

The construction of the coax port is a copy of the sample schematic shown in the CTI data sheet and the LERIC data sheet. The schematic shows no termination on the BNC connector. For this reason, the repeater can only be connected to a coax line as a center tap. If a  $50\Omega$  resistor was inserted between the lines connected to the BNC, the coax port could be used as the end of a coax network. JP19 connects a  $25\Omega$  resistor in parallel with the BNC connector thus providing termination when the coax port is disconnected from the network. The PM6512 provides the isolated  $-9V$  supply needed to properly operate the CTI.

### 3.4 Cascading

Unlike the DP83955, the DP83956's bus signals allow for external transceivers for cascading via ribbon cable. TTL drivers are used in this design. Open collector non-inverting bus drivers (74ALS1035) are used to drive the  $\overline{ACTN}$  and  $\overline{ANYXN}$  signals. A 4-bit transceiver (74ALS243) is used to drive the  $\overline{IRE}$ ,  $\overline{IRC}$ ,  $\overline{COLN}$ , and  $\overline{IRD}$  lines. A 14 pin header is used to connect boards via ribbon cable in a daisy chain. An input and output connector are positioned on the board for easy stacking and connecting. Since pull up resistors are only needed on one board when multiple boards are cascaded, a TRI-STATE<sup>®</sup> buffer (74ALS241) is used to control the connection of the pull up resistors. With no boards cascaded, the buffer pulls up all of the bus resistors to  $V_{OH}$ . Two boards can be cascaded by simply connecting the output connector of one board to the input connector of another board. The pull up resistors will only be active when the input connector is not connected to an output connector. In a chain of boards, this only occurs for the board at the top of the chain. The top board will have its pull up resistors pulled high, while the lower boards will have high impedance connected to their pull up resistors. Pin 1 of input connector will trigger the 74ALS241 to connect the pull up resistors to a high impedance output.

### 3.5 LED Display

The LED display on the DP83956EB-SA board consists of 4 LEDs that represent real time status for the hub. The LED status is continuously strobed out from the data bus of both LERIC controllers. *Figure 8* shows the logic required for a minimum mode display. Since the DP83956EB-SA contains two controllers, the hub status is obtained from both controllers. The status displayed on the board is jabber (packet longer than 5 ms), collision, reception, and port partitioned. A 74ALS874 is used to latch in the strobed data since its dual 4-bit design allows for separate clock signals. The status for two controllers are active low ORed by the 74ALS09 AND gate. The open collector output on the 74ALS09 allows for the construction of a pulse stretcher on the jabber signal. The 10 ms jabber signal is invisible to the user, so a pulse stretcher is necessary to make that signal visible. The Schmitt trigger inverters that are used for the output of the jabber signal, are in the same package as the inverters used in the mode lode logic, so no additional chips are required to make the pulse stretcher. The LERIC controllers hold the receive and collision lights for 30 ms or until the next activity. The any partition LED remains lit as long as any port on either of the LERICs is partitioned.

### 3.6 Mode Load Logic

The mode load (mload) logic (*Figure 9*) configures the two LERIC controllers on the board upon power-on and when the reset button is pushed. Many configuration bits are hard wired since the board only allows for one setting (see the DP83955/56 data sheet for more details on mload). Three configuration bits are accessible by the user by a three position DIP switch on the board. The user guide section explains these options in further detail. Both LERIC controllers on the board will get the same configuration as can be seen in the schematic. The board also contains a reset button for resetting the controllers and performing a mload. The

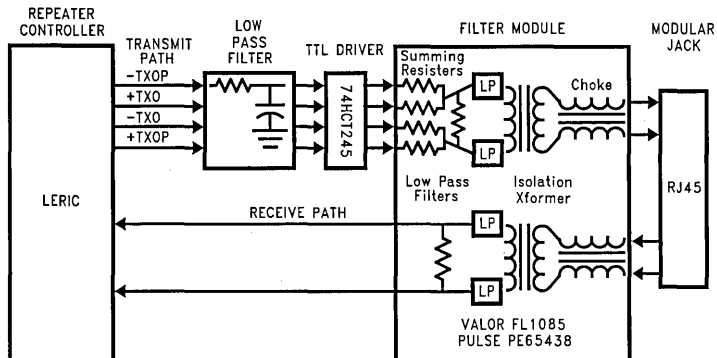


FIGURE 4. A Twisted Pair Interface for a Single Port

TL/F/11849-4

74HC14 is used to make a clean edge out of the discharging capacitor when reset is pushed and provide delay between the RESET signal to the LERIC controllers and the control of the buffers.

**3.7 LED Status Header**

This 32-pin connector was placed on the board to allow the user to build additional circuitry for displaying per port status, counting hub events, or accessing the LERIC registers.

**TABLE III. LED Status Header**

| JP# | 1                   | 2                   |                    |
|-----|---------------------|---------------------|--------------------|
| 3   | DA7                 | DB7                 | } Input/<br>Output |
| 4   | DA6                 | DB6                 |                    |
| 5   | DA5                 | DB5                 |                    |
| 6   | DA4                 | DB4                 |                    |
| 7   | DA3                 | DB3                 |                    |
| 8   | DA2                 | DB2                 |                    |
| 9   | DA1                 | DB1                 |                    |
| 10  | DA0                 | DB0                 |                    |
| 11  | $\overline{WR}A$    | $\overline{WR}B$    | } Input<br>Only    |
| 12  | $\overline{RD}A$    | $\overline{RD}B$    |                    |
| 13  | $\overline{BUFEN}A$ | $\overline{BUFEN}B$ | } Output<br>Only   |
| 14  | DFSA                | DFSB                |                    |
| 15  | $\overline{STR}A$   | $\overline{STR}B$   | } Output<br>Only   |
| 16  | MIN/MAX             | +5V                 |                    |
| 17  | GND                 | +5V                 |                    |
| 18  | GND                 | +5V                 |                    |

The DP83956EB-SA is made of two LERICs that are labeled LERIC A and LERIC B. LERIC A contains twisted pair ports 2-7 and CTI port, while LERIC B contains twisted pair ports 8-13 and AUI port 14. The data bus for both chips must be isolated from each other, since the LERICs will strobe out different status information on the data pins. To construct a per-port (max mode) LED display, the MIN/MAX mode pin (JP14 pin 1) should be tied to 5V. This will change both LERICs mode load configuration for max mode when the board is powered on or reset. The data strobed on the data

pins will now contain an address (D7-D5) and data (D4-D0) that will represent status for the individual ports. An addressable latch (74LS259) can be used to easily decode the status information and drive the LEDs. Table IV shows the port number mapping from the board to the specific LERIC chip port numbers. The LERIC data sheet shows a sample configuration for the max mode display.

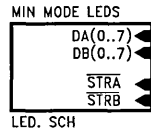
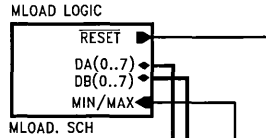
**TABLE IV. Port Number Conversion**

| LERIC Board Number | LERIC Port #     |
|--------------------|------------------|
| 1 — CTI            | Port 1 — LERIC A |
| 2                  | Port 2 — LERIC A |
| 3                  | Port 3 — LERIC A |
| 4                  | Port 4 — LERIC A |
| 5                  | Port 5 — LERIC A |
| 6                  | Port 6 — LERIC A |
| 7                  | Port 7 — LERIC A |
| 8                  | Port 2 — LERIC B |
| 9                  | Port 3 — LERIC B |
| 10                 | Port 4 — LERIC B |
| 11                 | Port 5 — LERIC B |
| 12                 | Port 6 — LERIC B |
| 13                 | Port 7 — LERIC B |
| 14 — AUI           | Port 1 — LERIC B |

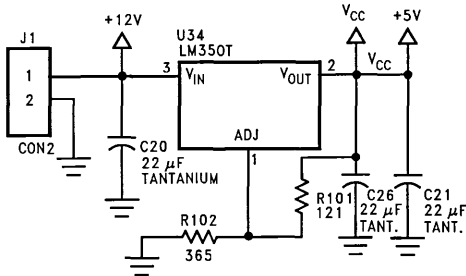
To construct event counters with the LED Status Header, the status should be decoded from the data strobed to the LEDs. Simple logic can then be trigger off the display freeze strobe (DFS) signal going true and the event being true. DFS will go high after activity has stopped on the repeater. DFS will remain high until the next activity or for a maximum of 30 ms. While DFS is high, the status for the last activity is held constant on the data bus. This is when the activity should be counted.

The construction of a register read or write module is possible since the  $\overline{RD}$  and  $\overline{WR}$  pins are available. The buffer enable signal is also included to allow for controlling buffers to perform read or writes. The LERIC data sheet contains more information on the signals needed to construct a register read/write module.

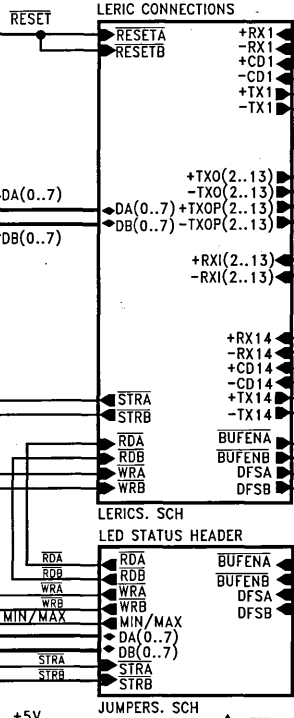
STARTUP & DISPLAY



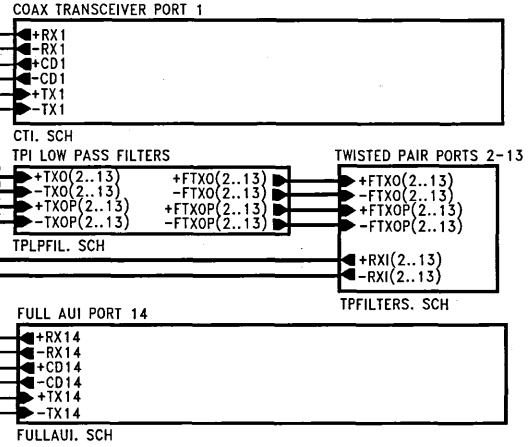
POWER CONNECTOR



LERICS & BUS LOGIC



PORT LOGIC



BYPASS CAPACITORS

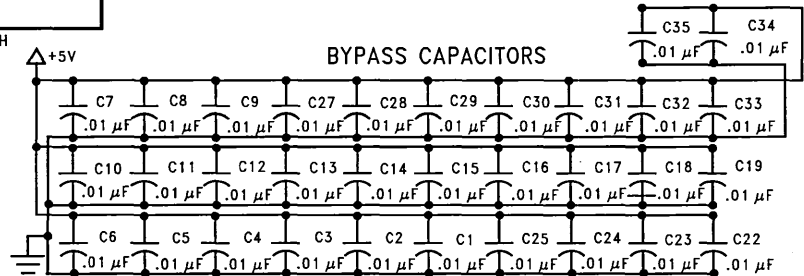


FIGURE 5. LERIC DP83956EB-SA Demonstration Board

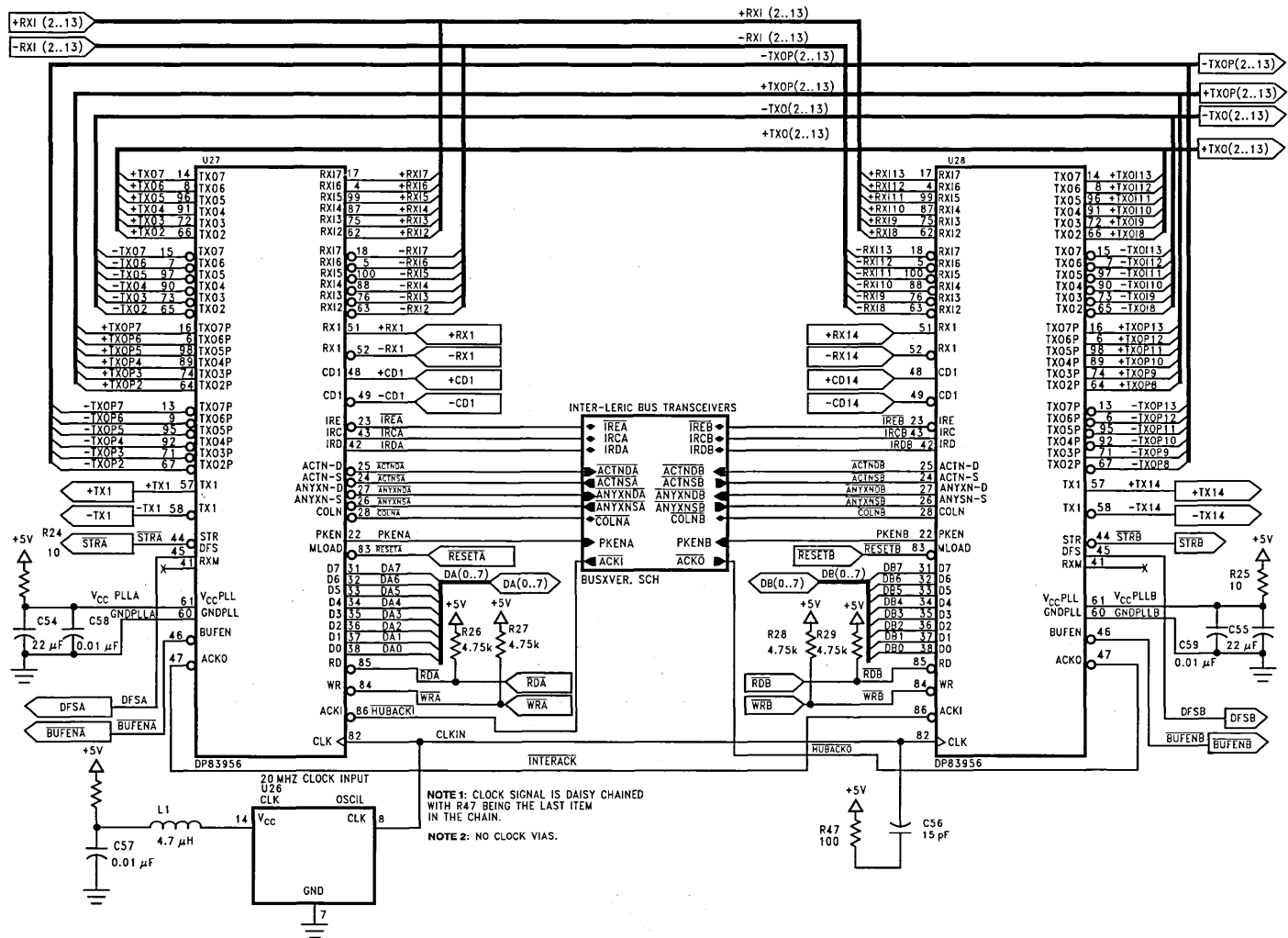
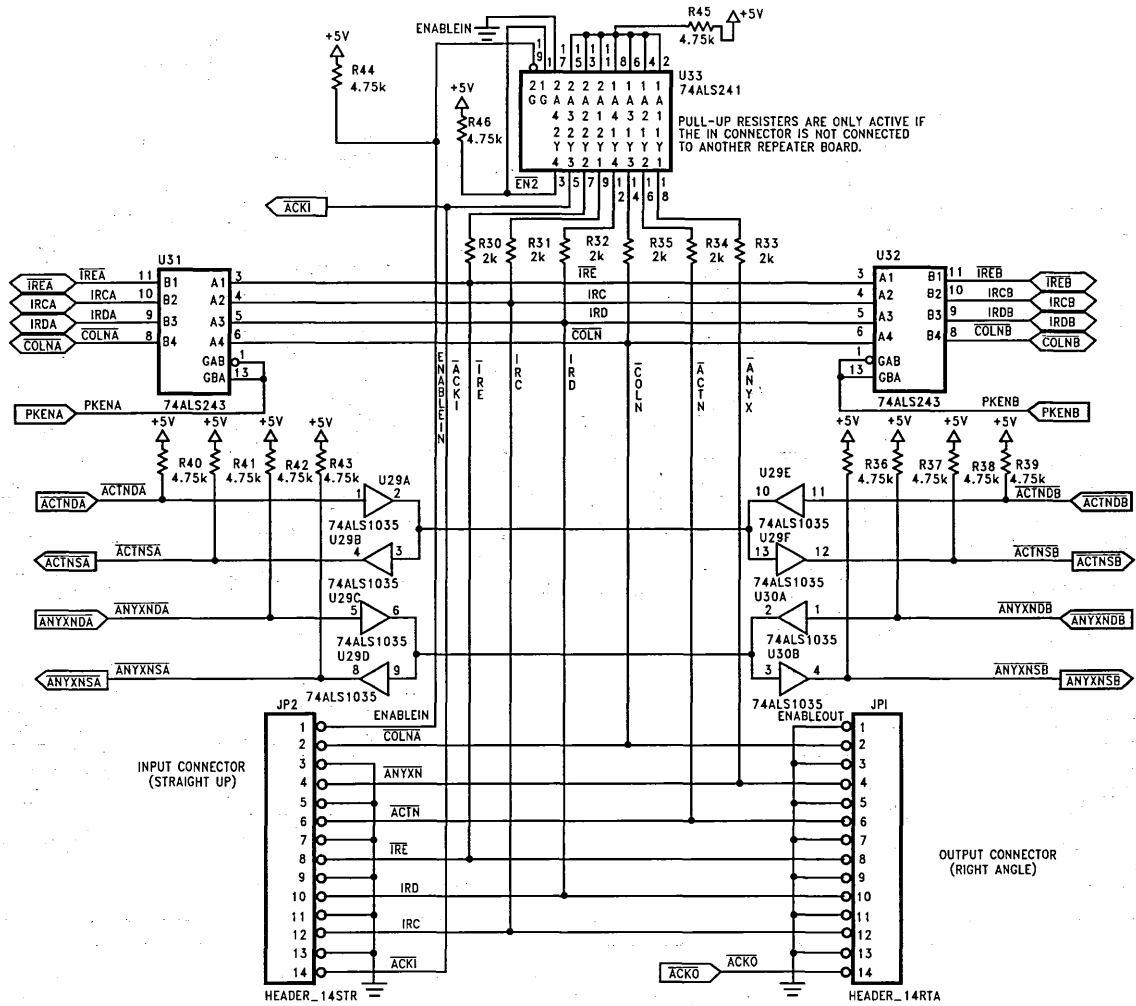


FIGURE 6. LERIC Connections





3-326

FIGURE 7. Inter-LERIC Bus Transceivers

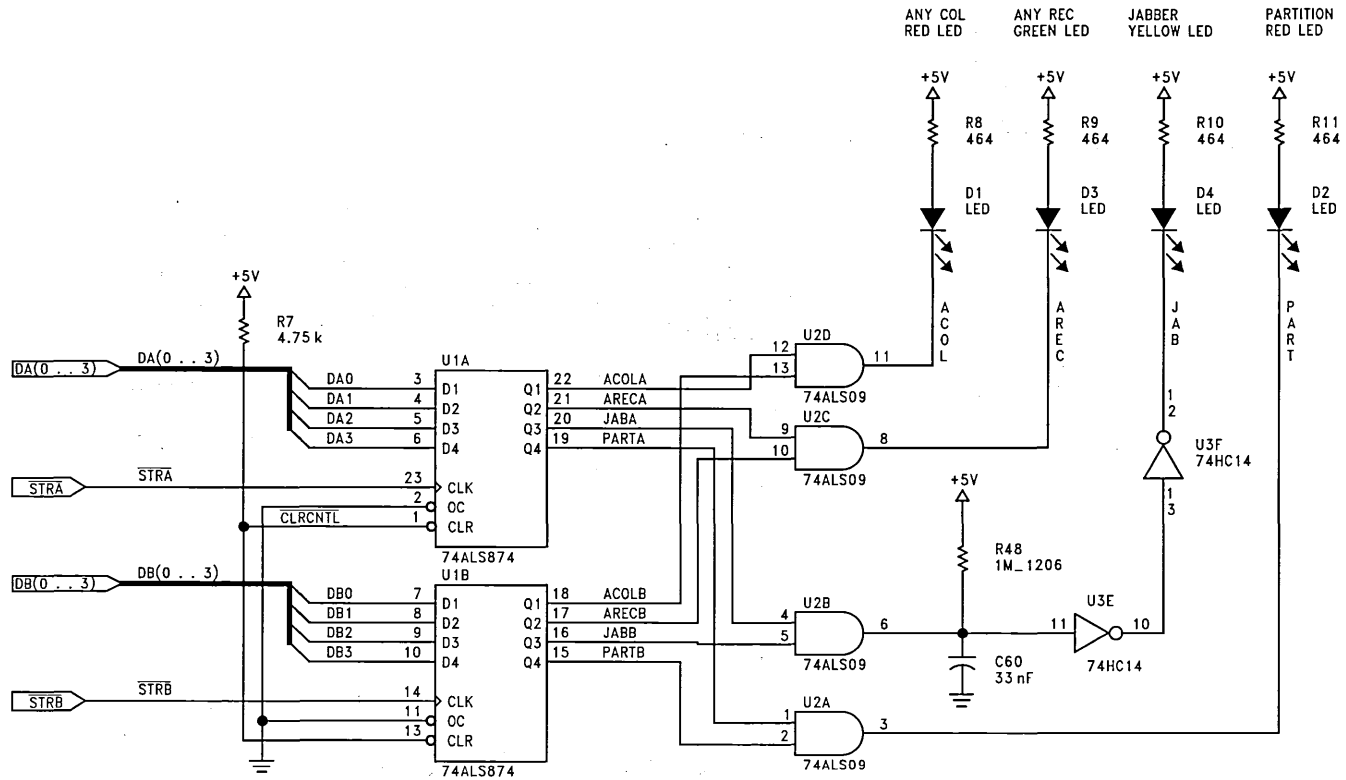


FIGURE 8. LERIC Hub Status Min Mode LEDs

TL/F/11849-8

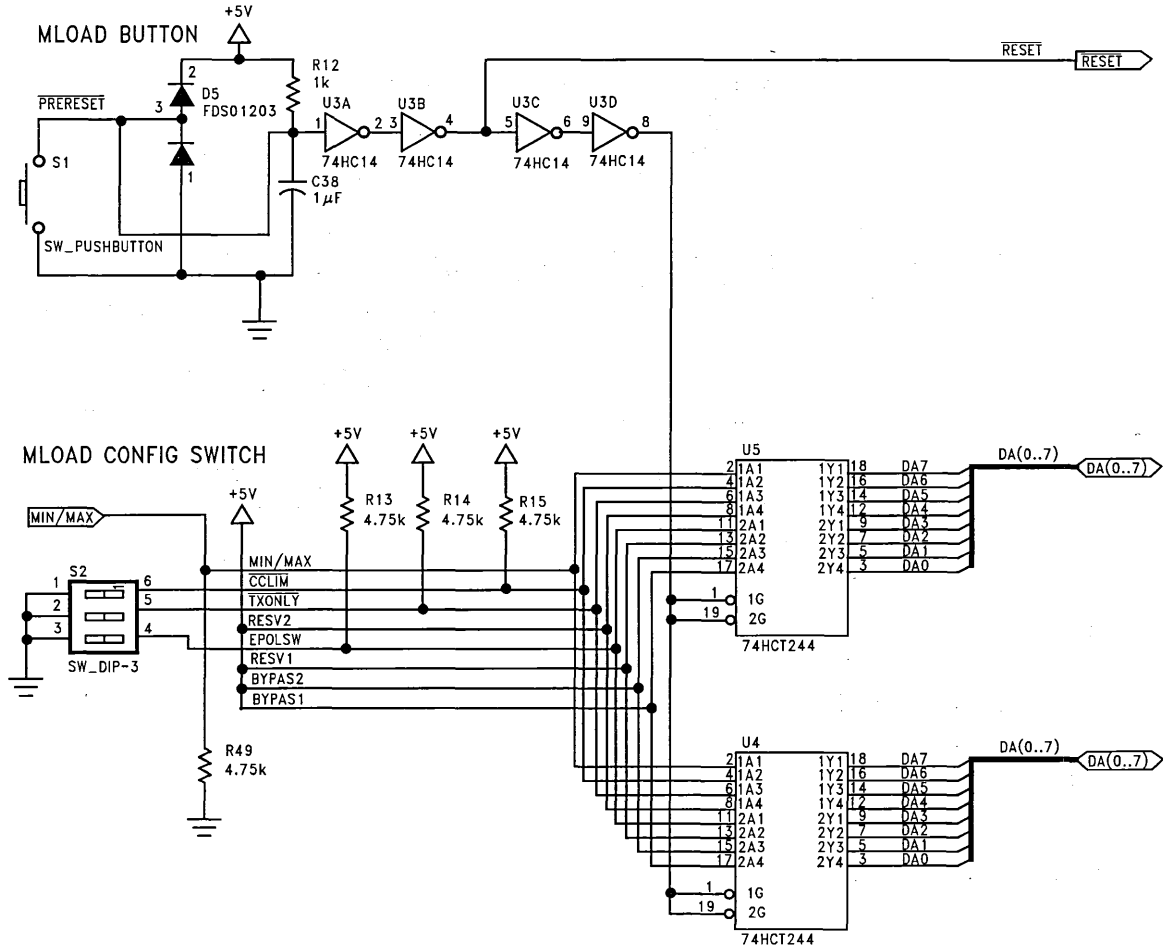


FIGURE 9. MLOAD Logic



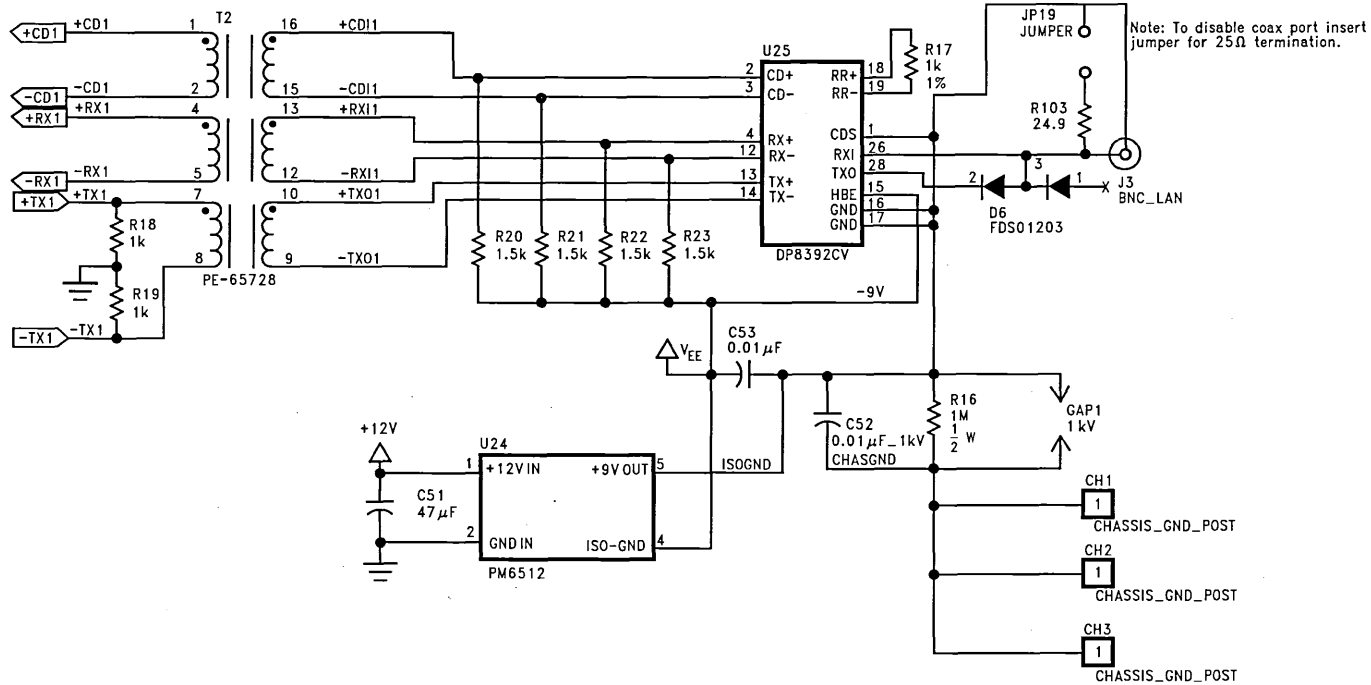


FIGURE 10. Coax Transceiver Port 1

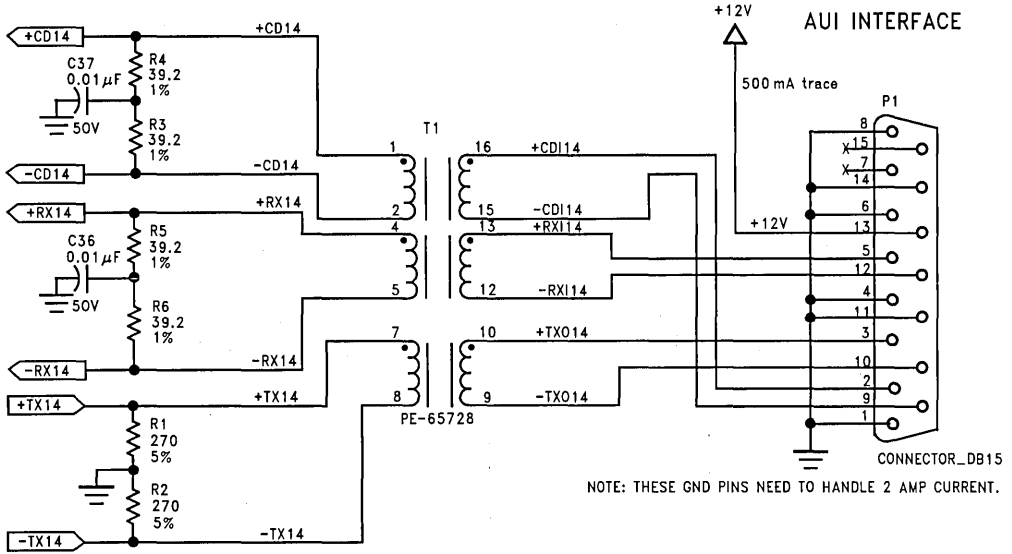


FIGURE 11. Full AUI Port 14

TL/F/11849-11

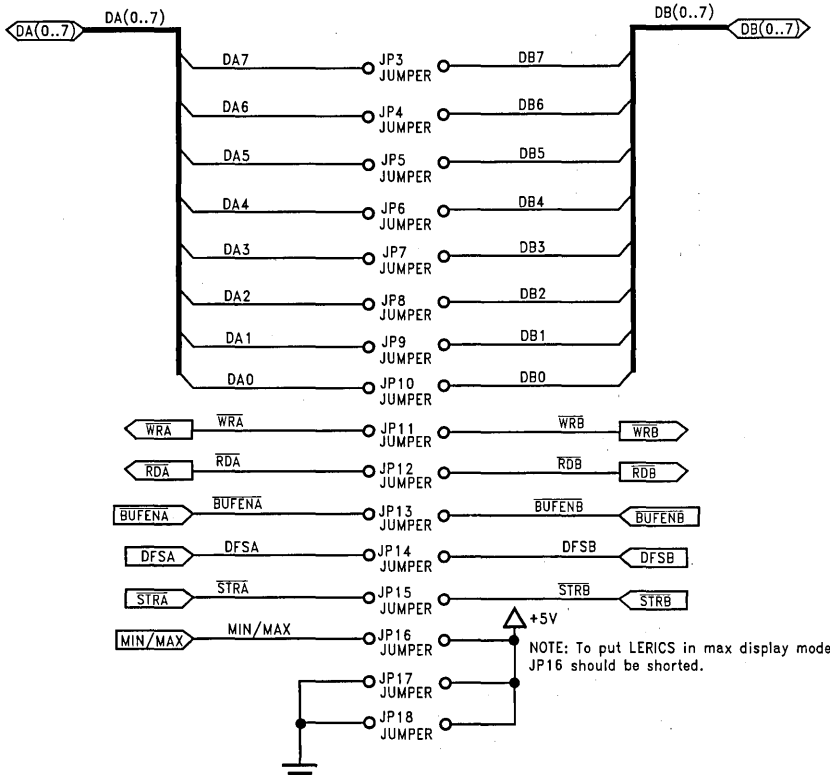


FIGURE 12. LED Status Header

TL/F/11849-12

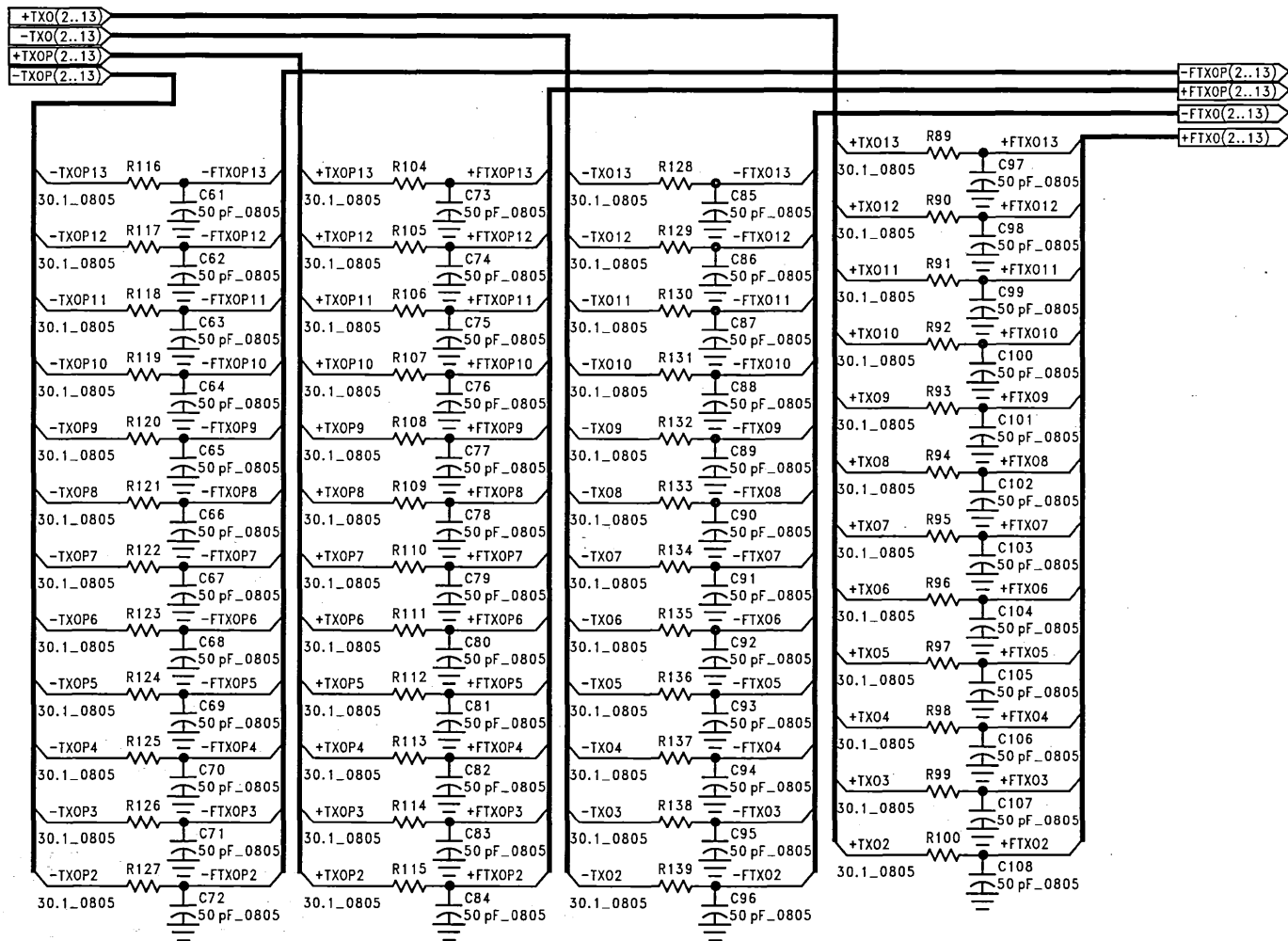
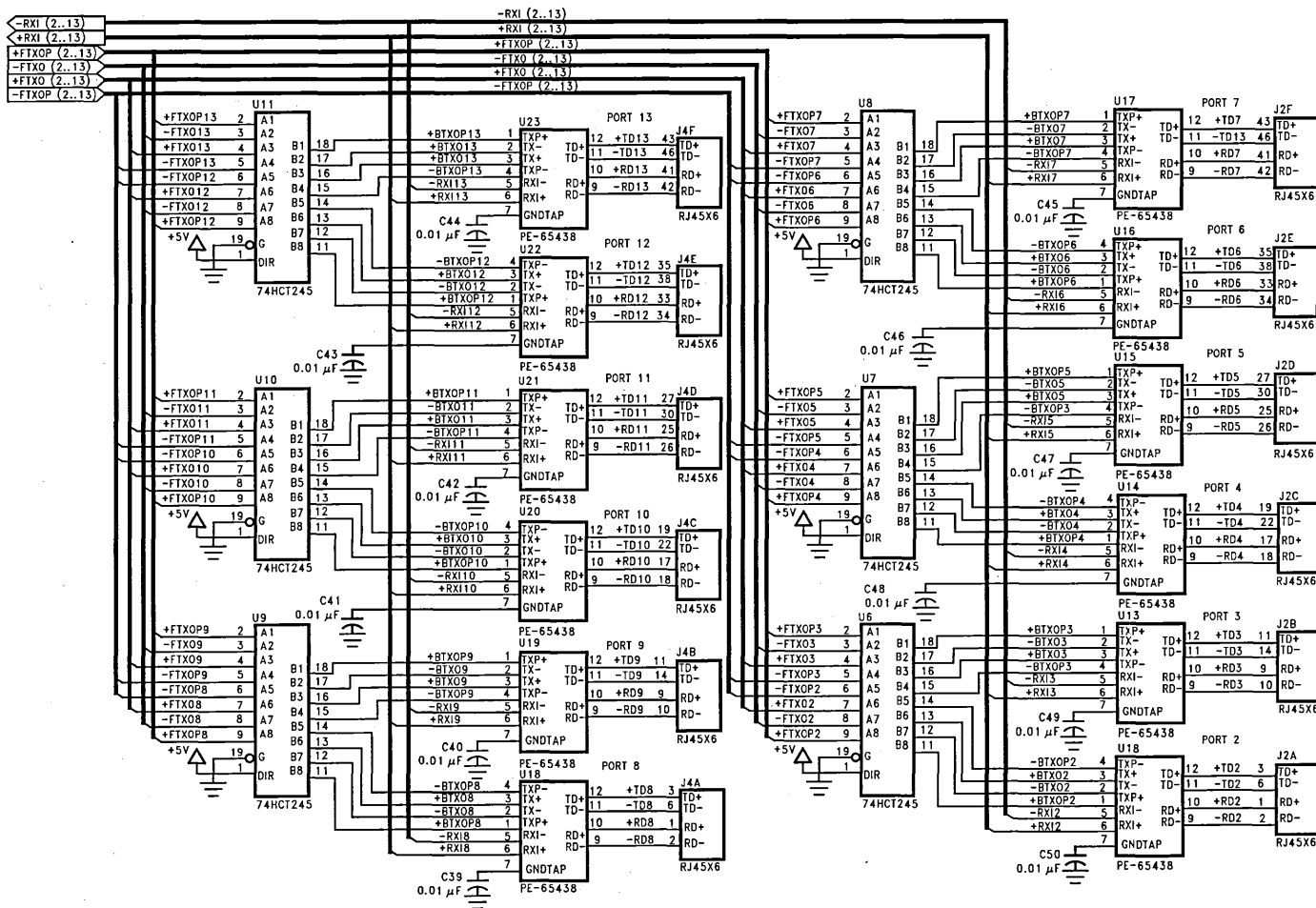


FIGURE 13.



3-332

FIGURE 14. Twisted Pair Ports 2-13

| 4.0 BILL OF MATERIALS |                                                                                      |                                                        |                                      |               |
|-----------------------|--------------------------------------------------------------------------------------|--------------------------------------------------------|--------------------------------------|---------------|
| Item #                | Reference (Location)                                                                 | Description                                            | Manufacturer and Manufacturer Part # | Qty. per Brd. |
| 1                     | U3                                                                                   | HC Inverter                                            | MM74HC14M                            | 1             |
| 2                     | U33                                                                                  | ALS Line Buffer/Driver Non-Invert                      | DM74ALS241AWM                        | 1             |
| 3                     | U31, U32                                                                             | ALS 4X Bus Transceiver—<br>Non-Inverting TRI-STATE     | DM74ALS243AM                         | 2             |
| 4                     | U29, U30                                                                             | ALS Hex Buffer Open Collector                          | DM74ALS1035M                         | 2             |
| 5                     | U4, U5                                                                               | HCT 8X Buffer/Driver TRI-STATE                         | MM74HCT244M                          | 2             |
| 6                     | U6 thru U11                                                                          | 8X Bus Transceiver—<br>Non-Inverting TRI-STATE         | MM74HCT245M                          | 6             |
| 7                     | U2                                                                                   | 4x2 in AND Gate—Open Collector                         | DM74ALS09M                           | 1             |
| 8                     | U1                                                                                   | Dual 4X D Latch                                        | DM74ALS874BWM                        | 1             |
| 9                     | U25                                                                                  | Ethernet Coaxial Transceiver, CTI™                     | DP8392CV                             | 1             |
| 10                    | U27, U28                                                                             | LERIC—100-Pin                                          | DP83956AVLY                          | 1             |
| 11                    | U34                                                                                  | Voltage Regulator—Variable—Hi Current                  | LM350T                               | 1             |
| 12                    | U24                                                                                  | 12V to 9V Converter                                    | PM6512                               | 1             |
| 13                    | T1, T2                                                                               | PULSE SMT AUI Transformer                              | PE-65728                             | 2             |
| 14                    | U12 thru U23                                                                         | TP Filter, Transformer and Summing Resistors           | Valor-FL1085                         | 12            |
| <b>DIODES/LEDS</b>    |                                                                                      |                                                        |                                      |               |
| 15                    | D5, D6                                                                               | Rect. 1V Dual SMD SOT 23                               | FDS1203                              | 2             |
| 16                    | D3                                                                                   | LED Green 3mm, 0.1 sp Diffused                         | IEE #LL231G                          | 1             |
| 17                    | D1, D2                                                                               | LED Red 3mm, 0.1 sp Diffused (hi eff.)                 | IEE #LL221R                          | 2             |
| 18                    | D4                                                                                   | LED Yellow 3mm, 0.1 sp Diffused                        | IEE #LL251Y                          | 1             |
| <b>CONNECTORS</b>     |                                                                                      |                                                        |                                      |               |
| 19                    | J3                                                                                   | CONN-BNC Pc/Mt, Low Profile Black                      | AMP-227161-9                         | 1             |
| 20                    | P1                                                                                   | CONN-DSUB Female 15P w/Solder Tab                      | AMP-747845-4                         | 1             |
| 21                    | JP2                                                                                  | CONN—14-Pin Header Straight—4 Wall                     | 3M-3598-6002                         | 1             |
| 22                    | JP1                                                                                  | CONN—14-Pin—Right Angle—4 Wall                         | 3M-3598-5002                         | 1             |
| 23                    | Cable Diagram                                                                        | CONN Receptacle—Center Polarized<br>Female—14 Position | 3M #3385 Series                      | 4             |
| 24                    | Cable Diagram                                                                        | Ribbon Cable—14 Conductor, 28AWG                       | 3M #3539/14                          | 1             |
| 25                    | J1                                                                                   | CONN PWR 2-Pin Angled—Male                             | AMP—#640389-2                        | 1             |
| 26                    | J2                                                                                   | CONN—RJ45 x 6—6 Port TP Conn Shielded                  | Stewart #SS-6688065                  | 2             |
| 27                    | JP3 thru JP18                                                                        | PIN Strip v/mt Brkway 16p 2 Row                        |                                      | 1             |
| 28                    | JP19                                                                                 | Pin Strip v/mt Brkway 2p 1 Row                         |                                      | 1             |
| 29                    | Assembly Diagram                                                                     | 1 Inch Aluminum Standoff—Hex Shape                     | Amatom #9743-A-0632                  | 4             |
| 29A                   | Assembly Diagram                                                                     | NUT #6-32 Hex Std S Steel                              |                                      | 4             |
| <b>CAPACITORS</b>     |                                                                                      |                                                        |                                      |               |
| 30                    | C1 thru C19,<br>C22 thru C25,<br>C27 thru C37,<br>C40 thru C50,<br>C53, C57 thru C59 | 0.01 $\mu$ F + 80 - 20 1206 50V C/C/SMD                |                                      | 50            |
| 32                    | C61 thru C108                                                                        | 47 pF $\pm$ 10% 0805 50V C/C/SMD                       |                                      | 48            |
| 33                    | C20, 21, 26, 54, 55                                                                  | 22 $\mu$ F $\pm$ 20% 7343 16V C/T/SMD                  |                                      | 5             |

## 4.0 BILL OF MATERIALS (Continued)

| Item #                        | Reference (Location)                                 | Description                             | Manufacturer and Manufacturer Part # | Qty. per Brd. |
|-------------------------------|------------------------------------------------------|-----------------------------------------|--------------------------------------|---------------|
| <b>CAPACITORS (Continued)</b> |                                                      |                                         |                                      |               |
| 34                            | C51                                                  | 47 $\mu$ F $\pm$ 20% 7343 10V C/T/SMD   |                                      | 1             |
| 35                            | C38                                                  | 1 $\mu$ F $\pm$ 20% 3216 16V C/T/SMD    |                                      | 1             |
| 36                            | C52                                                  | 0.01 $\mu$ F Radial $\pm$ 20% 7343 1 kV |                                      | 1             |
| 37                            | C56                                                  | 15 pF $\pm$ 10% 0805 50V C/C/SMD        |                                      | 1             |
| 38                            | C60                                                  | 0.33 $\mu$ F +80/-20 1206 50V C/C/SMD   |                                      | 1             |
| 39                            | GAP1                                                 | 0.75 pF Spark Gap 1 kV DC               |                                      | 1             |
| <b>MISC</b>                   |                                                      |                                         |                                      |               |
| 29B                           | Assembly Diagram                                     | Lock Washer #6 Ex-Tooth Std             |                                      | 5             |
| 29C                           | Assembly Diagram                                     | Washer #6 Std                           |                                      | 9             |
| 28A                           | JP19                                                 | Jumper Shunt 2p 1 Row 0.1 0.250 Hi      |                                      | 1             |
| <b>RESISTORS</b>              |                                                      |                                         |                                      |               |
| 40                            | R3 thru R6                                           | 39.2 R 1% 1206 $\frac{1}{8}$ W R/F/SM   |                                      | 4             |
| 41                            | R8 thru R11                                          | 464 R 1% 1206 $\frac{1}{8}$ W R/F/SM    |                                      | 4             |
| 42                            | R12, R17 thru R19                                    | 1.0k R 1% 1206 $\frac{1}{8}$ W R/F/SM   |                                      | 4             |
| 43                            | R20 thru R23                                         | 1.5k R 1% 1206 $\frac{1}{8}$ W R/F/SM   |                                      | 4             |
| 44                            | R30 thru R35                                         | 2.0k R 1% 1206 $\frac{1}{8}$ W R/F/SM   |                                      | 6             |
| 45                            | R7, R13 thru R15<br>R26 thru R29<br>R36 thru 46, R49 | 4.75k R 1% 1206 $\frac{1}{8}$ W R/F/SM  |                                      | 20            |
| 46                            | R1, R2                                               | 270 R 1% 1206 $\frac{1}{8}$ W R/F/SM    |                                      | 2             |
| 47                            | R48                                                  | 1.0M R 5% 1206 $\frac{1}{8}$ W R/F/SM   |                                      | 1             |
| 48                            | R16                                                  | 1.0M R 5% $\frac{1}{2}$ W Radial Comp   |                                      | 1             |
| 49                            | R24, R25                                             | 10 R 1% 1206 $\frac{1}{8}$ W R/F/SM     |                                      | 2             |
| 50                            | R89 thru R139                                        | 30.1 R 1% 0805 R/F/SM                   |                                      | 48            |
| 51                            | R101                                                 | 121 R 1% 1206 $\frac{1}{8}$ W R/F/SM    |                                      | 1             |
| 52                            | R102                                                 | 365 R 1% 1206 $\frac{1}{8}$ W R/F/SM    |                                      | 1             |
| 53                            | R103                                                 | 24.9 R 1% 1206 $\frac{1}{8}$ W R/F/SM   |                                      | 1             |
| 54                            | R47                                                  | 100 R 1% 1206 $\frac{1}{8}$ W R/F/SM    |                                      | 1             |

| 4.0 BILL OF MATERIALS (Continued) |                      |                                                                                                       |                                      |               |
|-----------------------------------|----------------------|-------------------------------------------------------------------------------------------------------|--------------------------------------|---------------|
| Item #                            | Reference (Location) | Description                                                                                           | Manufacturer and Manufacturer Part # | Qty. per Brd. |
| <b>INDUCTORS</b>                  |                      |                                                                                                       |                                      |               |
| 55                                | L1                   | 4.7 $\mu$ H/ $\pm$ 10% 1210 Induct/SMD                                                                |                                      | 1             |
| <b>SWITCHES</b>                   |                      |                                                                                                       |                                      |               |
| 56                                | S2                   | DIP Switch—3 Position Rocker Unsealed                                                                 | AMP-3-435166-0                       | 1             |
| 57                                | S1                   | Push Button—2 Pole N.O. Momentary                                                                     | Alco TP11CG-PC-0                     | 1             |
| <b>OSCILLATORS</b>                |                      |                                                                                                       |                                      |               |
| 58                                | U26                  | Crystal 20.0 MHz Osc 100 ppm 14p 4 Conn                                                               |                                      | 1             |
| 59                                | U34B                 | Screw—6-32 x 0.250 P/Head SS                                                                          |                                      | 1             |
| 60                                | U34C                 | Nut-Hex Std SS                                                                                        |                                      | 1             |
| 61                                | Assembly Diagram     | Std IDT Connector—0.156C<br>Double Cantalever Contact<br>Crimp Style Contact                          | MOLEX: 09-06-5027                    | 1             |
| 62                                | Stancor              | Power Supply—Wall Mnt. AC Adaptor<br>+ 12 @1A Unregulated<br>Plug Compatible: Switchcraft<br>#PC-722A | Stancor:<br>STA-4812A                | 1             |







Section 4

ETHERNET PROTOCOL  
PRODUCTS

**100 Mb/s Fast Ethernet  
Protocol Products**



## Section 4 Contents

|                                                                          |       |
|--------------------------------------------------------------------------|-------|
| DP83223A TWISTER High Speed Networking Transceiver Device .....          | 4-3   |
| DP83840 10/100 Mb/s Ethernet Physical Layer .....                        | 4-21  |
| DP83850 100 Mb/s Repeater Interface Controller .....                     | 4-76  |
| DP83856 100 Mb/s Repeater Information Base .....                         | 4-109 |
| AN-986 An Introduction to Auto-Negotiation .....                         | 4-137 |
| AN-1010 100BASE-TX Unmanaged Repeater Design Recommendations .....       | 4-144 |
| AN-336 Understanding Integrated Circuit Package Power Capabilities ..... | 4-169 |
| AN-780 Operating ECL from a Single Positive Supply .....                 | 4-174 |
| Transmission Line Concepts .....                                         | 4-181 |
| System Considerations .....                                              | 4-194 |

# DP83223A TWISTER™ High Speed Networking Transceiver Device

## General Description

The DP83223A Twisted Pair Transceiver is an integrated circuit capable of driving and receiving either binary or MLT-3 encoded datastreams. The DP83223A Transceiver is designed to interface directly with standards compliant FDDI and 100BASE-TX chip sets allowing low cost data links over copper based media. The DP83223A allows links of up to 100 meters over both Shielded Twisted Pair (STP) and datagrade Unshielded Twisted Pair (UTP) or equivalent. The electrical performance of the DP83223A meets or exceeds all performance parameters specified in the ANSI X3T12 TP-PMD draft standard and the IEEE 802.3 100BASE-TX Fast Ethernet Specification. The DP83223A also provides important features such as baseline restoration, TRI-STATE® capable transmit outputs, and controlled transmit output edge rates.

## Features

- Compliant with ANSI X3T12 TP-PMD draft standard
- Compliant with IEEE 802.3 100BASE-TX Ethernet draft standard
- Integrated baseline restoration circuit
- Integrated transmitter and receiver with adaptive equalization circuit
- Programmable binary or MLT-3 operation
- Isolated TX and RX power supplies for minimum noise coupling
- Controlled transmit output edge rates for reduced EMI
- Tri-State capable current transmit outputs
- Loopback feature for board diagnostics
- Programmable transmit voltage amplitude

## Block Diagram

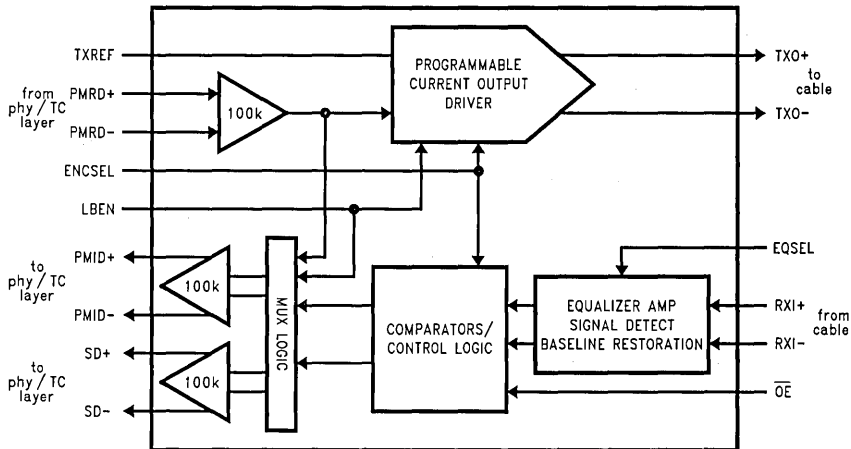


FIGURE 1. DP83223A Transceiver Block Diagram

TL/F/11886-1

# 1.0 Functional Description

## OVERVIEW

The DP83223A TWISTER is an enhanced version of the DP83223 transceiver device. The new DP83223A device has been designed to be backward compatible with the existing DP82223 device such that current PMD designs based on the DP83223 will accept, without modification, the new DP83223A device.

The DP83223A consists of seven major functional blocks as shown in Figure 1. The Transmit section consists of a 100K ECL input buffer for PMRD± and the Programmable Current Output Driver. The Programmable Current Output Driver can be configured to convert the incoming binary (NRZ or NRZI) datastream to a current sourced MLT-3 encoded datastream or current sourced binary datastream depending on the state of the ENCSEL input pin.

The Receive section consists of the following functional blocks: a differential input Equalization Amplifier with Signal Detect and baseline restoration circuitry, signal Comparators with Control Logic, Loopback Multiplexer Logic, and differential 100K ECL output drivers for PMID and Signal Detect.

In adaptive or full equalization mode, as selected by the EQSEL input pin, the receive data is first equalized and then amplified for signal detection. If the receive equalizer is turned off, the data is then only amplified for signal detection. In either case, the average baseline of the incoming signal is continuously monitored and restored given any baseline wander that may occur.

The Comparator/Control Logic block performs several functions. Primarily, the comparators either quantize and decode incoming MLT-3 into binary or simply quantize the incoming binary signal depending on the state of the ENCSEL input. The control logic receives input from OE and ENCSEL enabling final signal detect indication and control of data regeneration.

The Loopback Multiplexer logic performs the function of routing the transmit data at the PMRD± inputs to either the PMID± pins (loopback enabled) or to the TXO± current outputs (normal operation).

Finally, 100K ECL output drivers are used to drive both PMID± receive data and SD± Signal Detect data to the appropriate clock recovery circuit.

### 1.1 ENCODE/DECODE

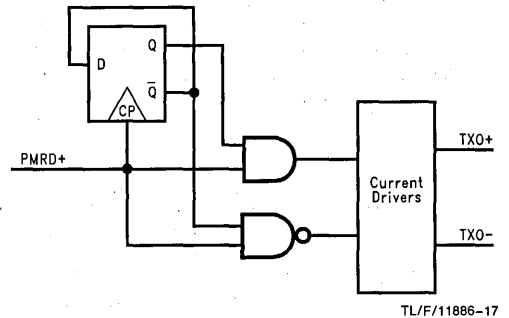
The PMRD± inputs of the DP83223A TWISTER will accept either NRZ or NRZI PECL binary up to 155 Mbps. With the ENCSEL (Encode Select) input pin forced low, binary data is asynchronously encoded to a current sourced MLT-3 data stream for transmission. For twisted pair FDDI or 100BASE-TX implementations, NRZI data (as provided by the Physical Layer) is required.

With ENCSEL forced high, incoming binary data is not encoded but is directly converted to a current sourced binary datastream.

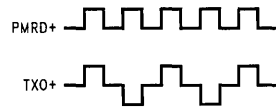
The receiver function of the DP83223A TWISTER is also determined by the state of the ENCSEL pin. With ENCSEL forced low, the receiver will decode the incoming MLT-3

data stream to the original binary version. This decoded binary signal is converted to PECL levels and then routed to the PMID± outputs. With ENCSEL forced high, the incoming binary data is regenerated, converted to PECL levels and routed to the PMID± outputs.

The process of encoding binary to MLT-3 and decoding MLT-3 back to binary is illustrated in the simplified logic circuits and associated bit patterns given in Figures 2 and 3.

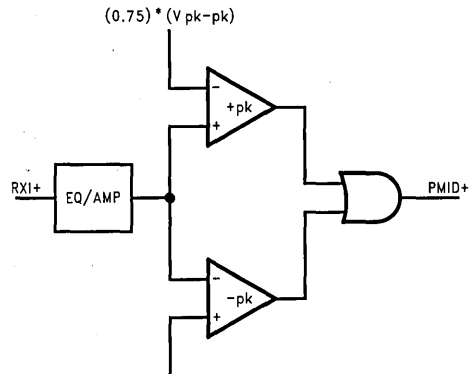


TL/F/11886-17

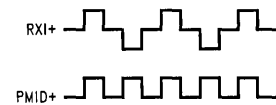


TL/F/11886-18

FIGURE 2. Binary to MLT-3 Encoding



TL/F/11886-19



TL/F/11886-20

FIGURE 3. MLT-3 to Binary Decoding

## 1.0 Functional Description (Continued)

### 1.2 BASELINE RESTORATION

The DP83223A TWISTER has been designed to correct for baseline wander as defined in the ANSI FDDI TP-PMD specification. Baseline wander can generally be defined as the change in the average DC content, over time, of an AC coupled digital transmission over a given transmission medium. (i.e., copper wire)

Baseline wander results from the interaction between the low frequency components of a bit stream being transmitted and the frequency response of the AC coupling component(s) within the transmission system. If the low frequency content of the digital bit stream goes below the low frequency pole of the AC coupling transformers then the droop characteristics of the transformers will dominate resulting in potentially serious baseline wander.

The digital oscilloscope plot provided in *Figure 4* illustrates the severity of the baseline wander event generated by the frame specified in the ANSI FDDI TP-PMD standard. This event consists of a DC offset of approximately 750 mV with a period of approximately 360  $\mu$ s. Left uncompensated, this event would cause many bit errors which could potentially isolate the receiving node from the FDDI network.

It is interesting to note that the probability of a baseline wander event serious enough to corrupt data is very low. In fact, it is reasonable to virtually bound the occurrence of a baseline wander event serious enough to cause bit errors to a legal but premeditated, artificially constructed bit sequence loaded into the original MAC frame. Several studies have been conducted to evaluate the probability of various baseline wander events for FDDI transmission over copper. Contact the X3T12 ANSI group for further information.

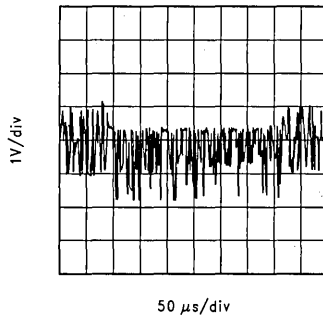


FIGURE 4. Baseline Wander Event

TL/F/11886-21

### 1.3 TRANSMIT AMPLITUDE CONTROL

The transmit amplitude of the signal presented at the TXO output pins can be controlled by varying the value of resistance between TXREF and TXGND. This TXREF resistor,  $R_{REF}$ , sets up a reference current which determines the final output current at TXO  $\pm$ . The DP83223A TWISTER will also automatically set the TXREF scaling factor to provide the correct transmit amplitude based on the selected mode of operation (MLT-3 or binary) without having to change the value of the TXREF resistor.

For 100 $\Omega$  Category-5 UTP cable implementations, the value of the TXREF resistor ( $R_{REF}$ ) is currently set at 510 $\Omega$ . This will yield either a 2  $V_{pk-pk}$  differential transmit amplitude for MLT-3 or a 1  $V_{pk-pk}$  differential transmit amplitude for binary. The following equations are useful for calculating  $R_{REF}$ :

$$R_{REF} = \frac{10.24 * Z_{CABLE}}{V_{OUT} (MLT-3)} \quad \text{or} \quad R_{REF} = \frac{5.12 * Z_{CABLE}}{V_{OUT} (binary)}$$

where:

- $R_{REF}$  is the TX amplitude reference resistor in Ohms ( $\Omega$ )
- $Z_{CABLE}$  is the characteristic differential impedance of the desired twisted pair cable ( $\Omega$ )
- $V_{OUT}$  is the differential peak-peak output voltage in Volts (V)
- 10.24 and 5.12 are related to the reference scaling factors

## 1.0 Functional Description (Continued)

### 1.4 TRANSITION TIME CONTROL

The transition times of the TXO± output signals are digitally synthesized resulting in closely matched and controlled rise and fall times. As with the previous TWISTER, the new DP83223A TWISTER incorporates this unique feature which significantly reduces the need for external filtering of the transmitted signal. The combination of these controlled edge rates and inexpensive magnetics (without additional filtering) generates clean rise and fall times compliant with the ANSI X3T12 TP-PMD draft standard and the IEEE 802.3 100BASE-TX Fast Ethernet Specification. In addition, control of the transmitter transition times reduces the EMI related board layout sensitivities. This offers a distinct advantage over transceivers dependent on external filtering for transition time control.

Figure 5 compares the power spectrum of two transmitters each driving a 1V binary scrambled HALT bitstream into a 100Ω load through a typical 350 μH 1:1 transformer. The power spectrum curve exhibiting lower relative power is from the DP83223A TWISTER transmitter with controlled output transition times. The curve exhibiting the higher relative power is from a transmitter which does not control the output transition time and therefore, would require additional external filtering to limit bandwidth and radiated emissions. The curves in Figure 6 make the same comparison for 2V MLT-3 transmission where the DP83223A TWISTER again exhibits lower relative power, eliminating the need for additional external transmit filtering.

### 1.5 ADAPTIVE EQUALIZATION

When transmitting data at high speeds over copper twisted pair cable, frequency dependent attenuation becomes a concern. In high speed twisted pair signalling the frequency content of the transmitted signal can vary greatly during normal operation based primarily on the randomness of the scrambled data stream. This variation in signal attenuation caused by frequency variations must be compensated for to ensure the integrity of the transmission.

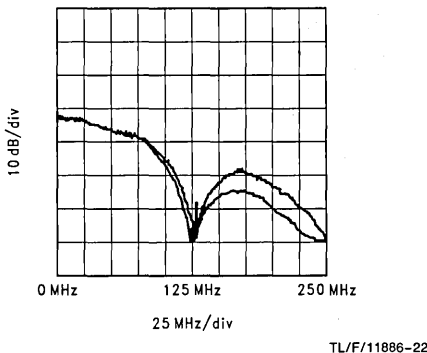
In order to ensure quality transmission when employing MLT-3 encoding, (and to a lesser extent, binary) the compensation must be able to adapt to various cable lengths and cable types depending on the installed environment. The selection of long cable lengths for a given implementation, requires significant compensation which will over-compensate for shorter, less attenuating lengths. Conversely, the selection of short or intermediate cable lengths requiring less compensation will cause serious under-compensation for longer length cables. Therefore, the compensation or equalization must be adaptive to ensure proper conditioning of the received signal regardless of the of the installed media.

In order to implement receiver adaptive equalization, a known relationship between transmit output amplitude and a receive input reference must be specified and controlled. Given these two parameters, the adaptive equalizer can determine the approximate cable length via signal attenuation at certain frequencies and actively compensate for cable variations.

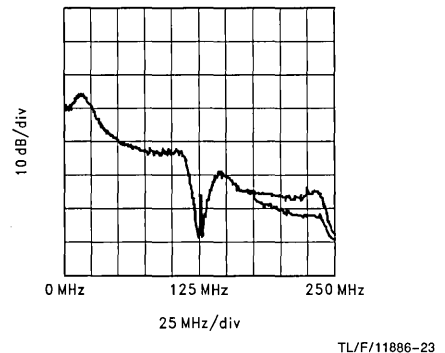
The curves given in Figure 7 provide insight to the actual attenuation at certain frequencies for given cable lengths. This is derived from the worst case frequency vs. attenuation figures as specified in the EIA/TIA Bulletin TSB-36. These curves indicate the significant variations in signal attenuation that must be compensated for by the receive adaptive equalization circuit.

As a measure of operation, Figures 8 through 13 indicate the performance and function of the adaptive equalizer in terms of jitter for a 2 V<sub>pk-pk</sub> differential MLT-3 signal.

Figure 8 represents a scrambled HALT, transmitted over 0 Meters of Category-5 cable as measured at the AII (Active Input Interface) of the receiver. Figure 9 represents the jitter of the recovered PECL data as measured differentially at the PMID± outputs. Figures 10 and 11 represent the performance over 50 Meters of Category-5 Cable and Figures 12 and 13 represent performance over 100 Meters of Category-5 cable.

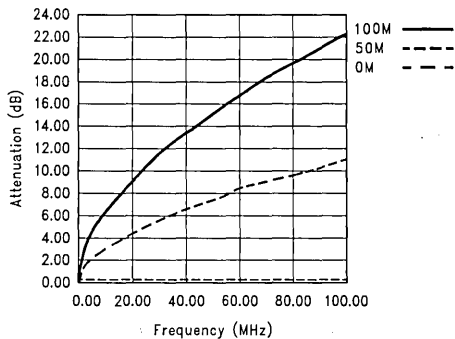


**FIGURE 5. 1V Binary Power Spectrum with and without Edge Rate Control**



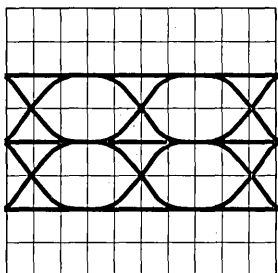
**FIGURE 6. 2V MLT-3 Power Spectrum with and without Edge Rate Control**

# 1.0 Functional Description (Continued)



**FIGURE 7. EIA/TIA Attenuation vs Frequency for 0, 50 and 100 Meters of CAT-5 Cable**

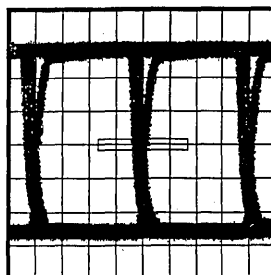
TL/F/11886-24



2 ns/div

TL/F/11886-25

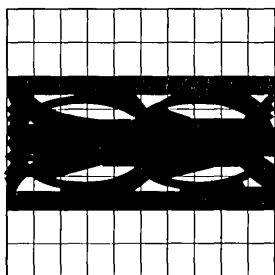
**FIGURE 8. MLT-3 Signal Measured at All after 0M of CAT-5 Cable**



2 ns/div

TL/F/11886-26

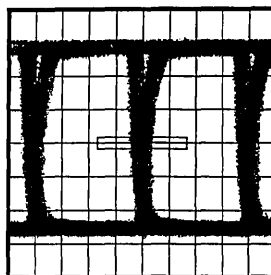
**FIGURE 9. Recovered ECL Peak Total Jitter for 0M of CAT-5 Cable (MLT-3) 660 ps**



2 ns/div

TL/F/11886-27

**FIGURE 10. MLT-3 Signal Measured at All after 50M of CAT-5 Cable**

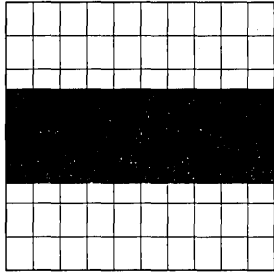


2 ns/div

TL/F/11886-28

**FIGURE 11. Recovered ECL Peak Total Jitter for 50M of CAT-5 Cable (MLT-3) 1.0 ns**

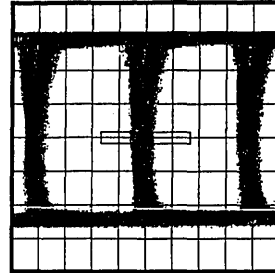
# 1.0 Functional Description (Continued)



2 ns/div

TL/F/11886-29

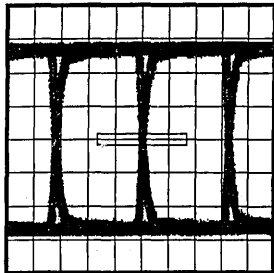
**FIGURE 12. MLT-3 Signal Measured at All after 100M of CAT-5 Cable**



2ns/div

TL/F/11886-30

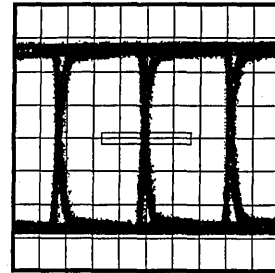
**FIGURE 13. Recovered ECL Peak Total Jitter for 100M of CAT-5 Cable (MLT-3) 1.08 ns**



2ns/div

TL/F/11886-31

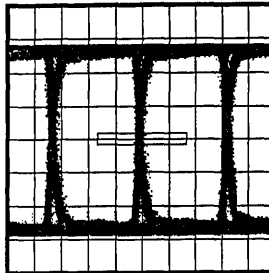
**FIGURE 14. Recovered ECL Peak Total Jitter for 0M of CAT-5 Cable (binary) 340 ps**



2ns/div

TL/F/11886-32

**FIGURE 15. Recovered ECL Peak Total Jitter for 50M of CAT-5 Cable (binary) 480 ps**



2ns/div

TL/F/11886-33

**FIGURE 16. Recovered ECL Peak Total Jitter for 100M of CAT-5 Cable (binary) 520 ps**



## 1.0 Functional Description (Continued)

### 1.6 JITTER PERFORMANCE

This section provides additional jitter performance measured using the DP83223A TWISTER transceiver. The TWISTER was configured to transmit a scrambled HALT data stream over various lengths of cable in both MLT-3 and binary modes of operation. The 0, 50 and 100 Meter plots provided in *Figures 9, 11 and 13* represent the MLT-3 mode total jitter performance.

*Figures 14, 15 and 16* indicate the jitter performance in the binary mode of operation also over 0, 50 and 100 Meter Category-5 cable lengths. All of these jitter results are given as the peak total jitter resulting from the combination of the transmitter, cable, and receiver.

### 1.7 SIGNAL DETECT

The signal detect function of the DP83223A TWISTER is incorporated to meet the specifications mandated by the ANSI FDDI TP-PMD Standard as well as the IEEE 802.3 100BASE-TX Standard.

The signal detect turn-on and turn-off thresholds for the TWISTER for both binary and MLT-3 modes of operation are illustrated in *Figure 17*. It is important to note that the signal detect function occurs after adaptive equalization and amplification of the received signal.

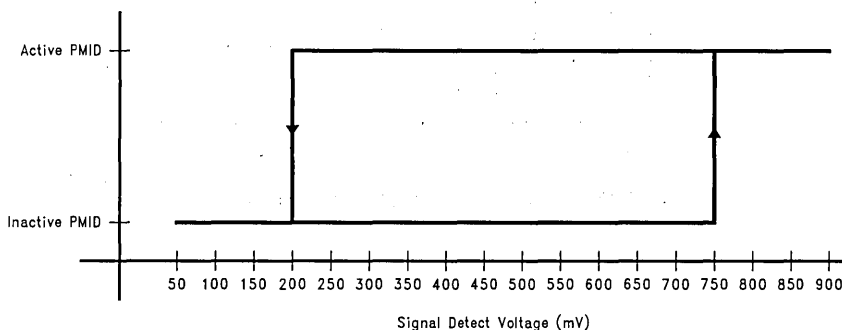


FIGURE 17. Signal Detect

TL/F/11886-34

### 1.8 OUTPUT ENABLE

The DP83223A TWISTER provides an output enable/disable function which is pin selectable via the Output Enable bar pin ( $\overline{OE}$ ). This pin was formerly labeled Cable Detect bar ( $\overline{CDET}$ ) on the previous DP83223 TWISTER. With  $\overline{OE}$  set to a logic high level, the TXO $\pm$  current outputs are both turned off. This causes the TXO outputs to effectively TRI-STATE regardless of the selected mode of operation (MLT-3 or Binary). Additionally, the PMID $\pm$  PECL outputs are forced to a steady state and the Signal Detect (SD+) output is forced to a logic low (SD+ = 0 and SD- = 1). With OE set to a logic low level, the TWISTER functions normally.

### 1.9 EQUALIZATION SELECT

The DP83223A TWISTER provides the added flexibility of controlling the type of receive equalization required for a given implementation. While adaptive equalization is the preferred method of cable compensation for TP-PMD FDDI and 100BASE-TX, the ability to switch the equalizer completely off or to a fixed maximum is provided.

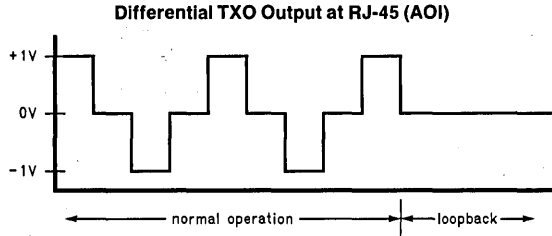
# 1.0 Functional Description (Continued)

## 1.10 LOOPBACK OPERATION

As with the previous TWISTER, the Loopback function of the DP83223A TWISTER (LBEN = 1) routes the signal at the PMRD inputs directly to the PMID outputs and forces SD+ high. However, the new TWISTER also forces the TXO± outputs to a true Quiet line state. This is accomplished by forcing the TXO± outputs to a balanced state

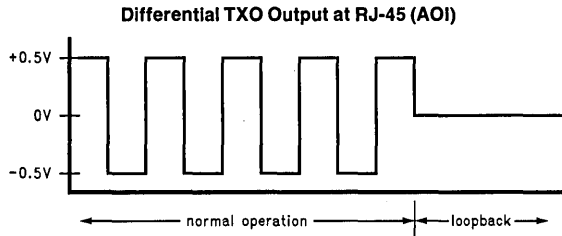
where each output sources 1/2 the peak current required for the selected mode of operation (MLT-3 or Binary).

Figure 18 illustrates the normal and true Quiet line signalling for normal and loopback functions while in the MLT-3 mode of operation. Figure 19 illustrates normal and loopback signalling for binary operation. Both figures are based on implementations using 100Ω Cat-5 UTP cable.



TL/F/11886-35

FIGURE 18. TXO True Quiet during Loopback (MLT-3 Mode)



TL/F/11886-36

FIGURE 19. TXO True Quiet during Loopback (Binary Mode)

## 2.0 System Connection Diagrams

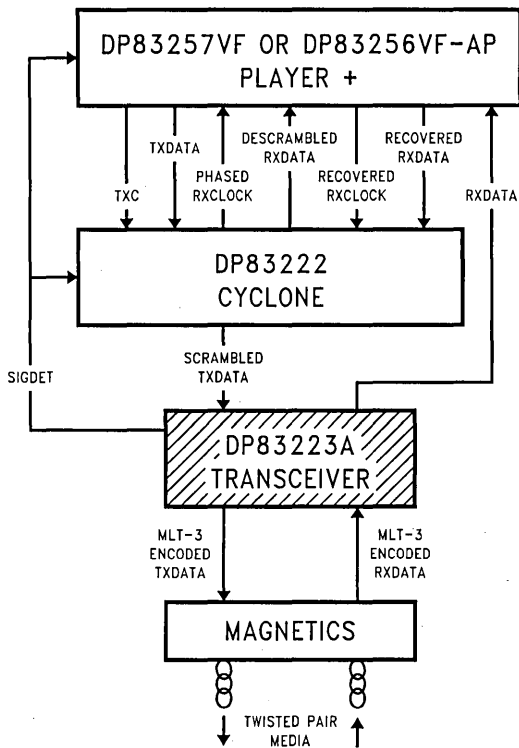


FIGURE 20. Typical TP-FDDI System Connection

TL/F/11886-37

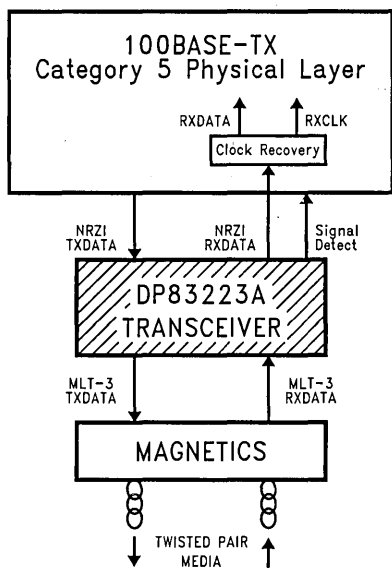
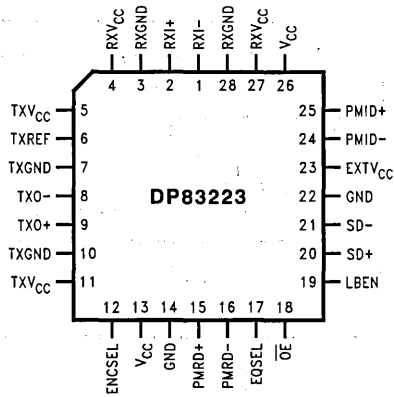


FIGURE 21. Typical 100BASE-TX System Connection

TL/F/11886-38

### 3.0 Pinout Information



28-Pin PLCC  
**FIGURE 23. Pin Configuration**

TL/F/11886-8

#### 3.1 PINOUT SUMMARY

| Signal | Pin No. | Description                  | Type                     |
|--------|---------|------------------------------|--------------------------|
| VCC    | 13, 26  | VCC                          | Supply                   |
| GND    | 14, 22  | GND                          | Supply                   |
| RXVCC  | 4, 27   | Receive VCC                  | Supply                   |
| RXGND  | 3, 28   | Receive GND                  | Supply                   |
| TXVCC  | 5, 11   | Transmit VCC                 | Supply                   |
| TXGND  | 7, 10   | Transmit GND                 | Supply                   |
| EXTVCC | 23      | External VCC                 | Supply                   |
| RXI±   | 2, 1    | Receive Data Inputs          | Differential Voltage In  |
| PMID±  | 25, 24  | Physical Media Indicate Data | ECL Out                  |
| PMRD±  | 15, 16  | Physical Media Request Data  | ECL In                   |
| TXO±   | 9, 8    | Transmit Data Outputs        | Differential Current Out |
| SD±    | 20, 21  | Signal Detect Outputs        | ECL Out                  |
| TXREF  | 6       | Transmit Amplitude Reference | Current Out              |
| ENCSEL | 12      | Encode Select Input          | CMOS In                  |
| LBEN   | 19      | Loopback Enable              | CMOS In                  |
| EQSEL  | 17      | Equalization Select          | 3-Level Select           |
| OE     | 18      | Output Enable Bar            | CMOS Schmitt Trig In     |

### 3.0 Pinout Information (Continued)

#### 3.2 PIN DEFINITIONS

**V<sub>CC</sub> (13, 26):** Positive power supply for the 100K ECL compatible circuitry. The Transceiver operates from a single +5 VDC power supply.

**GND (14, 22):** Return path for the 100K ECL compatible circuitry power supply.

**RXV<sub>CC</sub> (4, 27):** Positive power supply for the small signal receive circuitry. This power supply is intentionally separated from others to eliminate receive errors due to coupled supply noise.

**RXGND (3, 28):** Return path for the receive power supply circuitry. This power supply return is intentionally separated from others to eliminate receive errors due to coupled supply noise.

**TXV<sub>CC</sub> (5, 11):** Positive power supply required by the analog portion of the transmit circuitry. This power supply is intentionally separated from the others to prevent supply noise from coupling to the transmit outputs.

**TXGND (7, 10):** Return path for the analog transmit power supply circuitry. This supply return is intentionally separated from others to prevent supply noise from being coupled to the transmit outputs.

**EXTV<sub>CC</sub> (23):** Positive power supply for 100K ECL output circuitry.

**RXI $\pm$  (2, 1):** Balanced differential line receiver inputs. These inputs are capable of receiving either binary or MLT-3 encoded data.

**PMID $\pm$  (25, 24):** Differential 100K ECL compatible outputs which source the recovered receive-data to the appropriate clock recovery circuit.

**PMRD $\pm$  (15, 16):** Differential 100K ECL compatible inputs which accept transmit-data from the appropriate Physical Layer device

**TXO $\pm$  (9, 8):** Differential current driver outputs which drive either binary or MLT-3 encoded data over either 100 $\Omega$  or 150 $\Omega$  twisted pair cable. These outputs provide controlled rise and fall times designed to filter the transmitters output, reducing any associated EMI.

**SD $\pm$  (20, 21):** Differential 100K ECL compatible Signal Detect outputs which indicate that either a valid signal is present at the RXI $\pm$  inputs or that Loopback mode has been selected.

**TXREF (6):** Reference current pin allowing adjustment of TXO $\pm$  transmit amplitude. By placing a resistor between this pin and TXGND, a reference current is setup which results in a given transmit amplitude for a given application. Refer to section 1.3 of the Functional Description for reference current equations.

**ENCSEL (12):** The TTL compatible CMOS Encode Select input controls the encoded state of the signal at the TXO $\pm$  outputs. A logic high level at this input causes the TXO $\pm$  pins to output binary code and configures the receiver to receive binary. A logic low level causes the TXO outputs to source MLT-3 encoded data and configures the receiver to accept MLT-3 encoded data.

**LBEN (19):** TTL compatible CMOS Loopback Enable input pin selects the internal loopback path which routes the PMRD $\pm$  data to the PMID $\pm$  differential outputs and forces Signal Detect true. During loopback operation (LBEN = 1) the TXO $\pm$  outputs source a true QUIT onto the cable. Normal operation occurs when LBEN forced low.

**EQSEL (17):** This three level Equalization Select input controls the mode of receiver equalization. Forcing a median voltage level, accomplished by allowing EQSEL to float, selects the adaptive equalization mode which automatically regulates the equalization effects based on signal degradation caused by the media. Forcing a voltage less than  $\frac{1}{3}$  of V<sub>CC</sub>, selects full equalization which provides fixed equalization for a maximum length of cable. Forcing a voltage greater than  $\frac{2}{3}$  of V<sub>CC</sub> turns the receive equalizer off.

**$\overline{\text{OE}}$  (18):** The Schmitt Trigger Output Enable bar input is provided to support the option of true transmit disable. With  $\overline{\text{OE}}$  high, the signal detect output is forced low (SD+ = 0) which inhibits data reception by the PHY and the PMID outputs are forced to 100K ECL static levels (PMID+ = 0, PMID- = 1). The exception is in the case of Loopback when the Signal Detect output is forced high (SD+ = 1) regardless of all other conditions. With  $\overline{\text{OE}}$  low, the transceiver functions normally.

## 4.0 Functional Truth Tables

### 4.1 OUTPUT ENABLE

Output Enable

| $\overline{OE}$ | TXO Outputs     | PMID+ /PMID-    | SD+ /SD-        |
|-----------------|-----------------|-----------------|-----------------|
| 0               | Normal Function | Normal Function | Normal Function |
| 1               | TRI-STATE       | 0/1             | 0/1             |

### 4.2 EQUALIZATION SELECT

Equalization Select

| EQSEL  | Mode        |
|--------|-------------|
| < 1.5V | Full EQ     |
| Float  | Adaptive EQ |
| > 3.0V | EQ Off      |

### 4.3 ENCODE SELECT

Encode Select

| ENCSEL | Mode                       |
|--------|----------------------------|
| 0      | MLT-3 (TP-FDDI/100BASE-TX) |
| 1      | Binary                     |

### 4.4 LOOPBACK SELECT

Loopback Select

| LBEN | TXO Outputs     | PMID Outputs    | SD+ /SD-        |
|------|-----------------|-----------------|-----------------|
| 0    | Normal Function | Normal Function | Normal Function |
| 1    | True Quiet      | follow PMRD     | 1/0             |

The Loopback Function is intended for board diagnostics. In Loopback mode, the ECL signal applied to the PMRD inputs will appear at the PMID outputs regardless of the signal present at the RXI inputs.

## 5.0 Electrical Characteristics

### ABSOLUTE MAXIMUM RATINGS

| Symbol               | Parameter                | Conditions          | Min  | Typ | Max  | Units |
|----------------------|--------------------------|---------------------|------|-----|------|-------|
| V <sub>CC</sub>      | Logic Power              | Referenced to GND   | -0.5 |     | 7.0  | V     |
| RXV <sub>CC</sub>    | Received Power           | Referenced to RXGND | -0.5 |     | 7.0  | V     |
| TXV <sub>CC</sub>    | Transmit Power           | Referenced to TXGND | -0.5 |     | 7.0  | V     |
| EXTV <sub>CC</sub>   | ECL Output Power         | Referenced to GND   | -0.5 |     | 7.0  | V     |
| I <sub>ECL</sub>     | DC Output Current (High) |                     |      |     | -50  | mA    |
| ESD                  |                          |                     |      |     | 2    | KV    |
| T <sub>storage</sub> | Storage Temperature      |                     | -65  |     | +150 | °C    |

### 5.1 RECOMMENDED OPERATING CONDITIONS

| Symbol          | Parameter                | Conditions | Min  | Typ | Max  | Units |
|-----------------|--------------------------|------------|------|-----|------|-------|
| V <sub>CC</sub> | Supply Voltage           |            | 4.50 | 5.0 | 5.50 | V     |
| T <sub>A</sub>  | Operating Temperature    |            | 0    | 25  | 70   | °C    |
| PD              | Device Power Dissipation |            |      | 700 |      | mW    |

### 5.2 DC ELECTRICAL CHARACTERISTICS T<sub>A</sub> = 25°C, V<sub>CC</sub> = 5.0V

| Symbol                 | Parameter                | Conditions                                  | Min                    | Typ                  | Max                    | Units |
|------------------------|--------------------------|---------------------------------------------|------------------------|----------------------|------------------------|-------|
| I <sub>IHt</sub>       | TTL High Level Input     |                                             |                        |                      | 10                     | μA    |
| I <sub>ILt</sub>       | TTL Low Level Input      |                                             |                        |                      | -10                    | μA    |
| I <sub>IHschmitt</sub> | Schmitt High Level Input |                                             |                        |                      | 10                     | μA    |
| I <sub>ILschmitt</sub> | Schmitt Low Level Input  |                                             |                        |                      | -10                    | μA    |
| I <sub>IHeqsel</sub>   | EQSEL High Level Input   | Figure 24a                                  |                        |                      | 1000                   | μA    |
| I <sub>ILeqsel</sub>   | EQSEL Low Level Input    | Figure 24a                                  |                        |                      | -1000                  | μA    |
| I <sub>IHe</sub>       | ECL High Level Input     |                                             |                        |                      | 50                     | μA    |
| I <sub>ILe</sub>       | ECL Low Level Input      |                                             |                        |                      | 1.0                    | μA    |
| V <sub>IHt</sub>       | TTL High Level Input     |                                             | 2.0                    |                      |                        | V     |
| V <sub>ILt</sub>       | TTL Low Level Input      |                                             |                        |                      | 0.8                    | V     |
| V <sub>IHschmitt</sub> | Schmitt High Level Input |                                             |                        | 3*V <sub>CC</sub> /4 |                        | V     |
| V <sub>ILschmitt</sub> | Schmitt Low Level Input  |                                             |                        | V <sub>CC</sub> /4   |                        | V     |
| V <sub>IHeqsel</sub>   | EQSEL High Level Input   |                                             |                        | 2*V <sub>CC</sub> /3 |                        | V     |
| V <sub>ILeqsel</sub>   | EQSEL Low Level Input    |                                             |                        | V <sub>CC</sub> /3   |                        | V     |
| V <sub>IMeqsel</sub>   | EQSEL Mid Level Input    |                                             |                        | V <sub>CC</sub> /2   |                        | V     |
| V <sub>IHe</sub>       | ECL High Level Input     |                                             | V <sub>CC</sub> - 1165 |                      | V <sub>CC</sub> - 880  | mV    |
| V <sub>ILe</sub>       | ECL Low Level Input      |                                             | V <sub>CC</sub> - 1810 |                      | V <sub>CC</sub> - 1475 | mV    |
| V <sub>OHe</sub>       | ECL High Level Output    | Figure 24c                                  | V <sub>CC</sub> - 1025 |                      | V <sub>CC</sub> - 880  | mV    |
| V <sub>OLe</sub>       | ECL Low Level Output     | Figure 24c                                  | V <sub>CC</sub> - 1810 |                      | V <sub>CC</sub> - 1620 | mV    |
| I <sub>CCext</sub>     | External Supply Current  | Figure 24c                                  |                        | 55                   |                        | mA    |
| I <sub>CCint1</sub>    | Internal Supply Current  | ENCSEL = 1, $\overline{OE}$ = 0, Figure 24c |                        | 115                  |                        | mA    |
| I <sub>CCint2</sub>    | Internal Supply Current  | ENCSEL = 0, $\overline{OE}$ = 0, Figure 24c |                        | 135                  |                        | mA    |
| I <sub>CCtotal1</sub>  | Total Supply Current     | ENCSEL = 1, $\overline{OE}$ = 0, Figure 24c |                        | 170                  |                        | mA    |
| I <sub>CCtotal2</sub>  | Total Supply Current     | ENCSEL = 0, $\overline{OE}$ = 0, Figure 24c |                        | 190                  |                        | mA    |

## 5.0 Electrical Characteristics (Continued)

5.2 DC ELECTRICAL CHARACTERISTICS  $T_A = 25^\circ\text{C}$ ,  $V_{CC} = 5.0\text{V}$  (Continued)

| Symbol             | Parameter                         | Conditions                                                                                                | Min | Typ   | Max | Units            |
|--------------------|-----------------------------------|-----------------------------------------------------------------------------------------------------------|-----|-------|-----|------------------|
| $I_{TXOmatch1}$    | Transmit Current Matching 1       | $LBEN = 1$ , $R_{REF} = 510\Omega$ , (Note 1)                                                             |     | 2.5   |     | %                |
| $I_{TXOmatch2}$    | Transmit Current Matching 2       | $ENCSEL = 0$ , $LBEN = 0$ , $R_{REF} = 510\Omega$<br>$PMRD+ = 0$ , $PMRD- = 1$ (Note 1)                   |     | 2.5   |     | %                |
| $I_{TXOleak\_tri}$ | TXO TRI-STATE Leakage Current     | $\overline{OE} = 1$ , $R_{REF} = 510\Omega$ , $V_{TXO-} = 0\text{V}$ ,<br>$V_{TXO+} = 0\text{V}$ (Note 2) |     |       | 10  | $\mu\text{A}$    |
| $TX_{ref\_mult1}$  | TXREF Multiplier for Binary       | $ENCSEL = 1$ , $R_{REF} = 510\Omega$ , Figure 24b,<br>(Note 3)                                            |     | 5.12  |     |                  |
| $TX_{ref\_mult2}$  | TXREF Multiplier for MLT-3        | $ENCSEL = 0$ , $R_{REF} = 510\Omega$ , Figure 24b<br>(Note 3)                                             |     | 10.24 |     |                  |
| $TX_{ref\_tol}$    | TXREF Multiplier Tolerance        | $ENCSEL = 1$ , $R_{REF} = 510\Omega$                                                                      |     | 2.5   |     | %                |
| $SD_{THon1}$       | Sig Det Turn-on Threshold 1       | $ENCSEL = 1$ , Figure 10 (Note 4)                                                                         |     | 750   |     | mV               |
| $SD_{THon2}$       | Sig Det Turn-on Threshold 2       | $ENCSEL = 0$ , Figure 10 (Note 4)                                                                         |     | 750   |     | mV               |
| $SD_{THoff1}$      | Sig Det Turn-off Threshold 1      | $ENCSEL = 1$ , Figure 10 (Note 4)                                                                         |     | 200   |     | mV               |
| $SD_{THoff2}$      | Sig Det Turn-off Threshold 2      | $ENCSEL = 0$ , Figure 10 (Note 4)                                                                         |     | 200   |     | mV               |
| $R_{INdiff}$       | RXI Differential Input Resistance |                                                                                                           |     | 8     |     | $\text{K}\Omega$ |

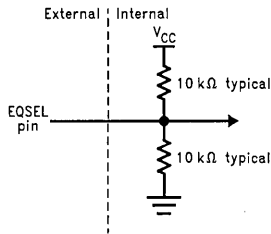
**Note 1:**  $I_{TXOmatch}$  refers to the percentage of mismatch in current between TXO+ and TXO- referenced to the total output current as defined by:

$$I_{TXOmatch} = \frac{I_{TXO+} - I_{TXO-}}{I_{TXO+} + I_{TXO-}} \times 100$$

**Note 2:**  $V_{TXO+}$  and  $V_{TXO-}$  refer to the voltage present at the TXO+ and TXO- output pins respectively.

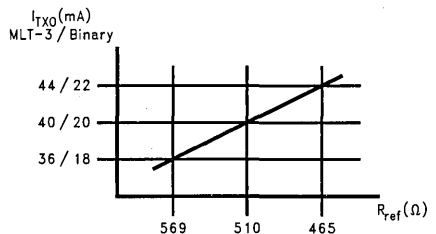
**Note 3:** Refer to Section 1.3 for further description of the TXREF Multiplier (scaling factor).

**Note 4:** Signal Detect turn on and turn off thresholds are measured differentially directly across the RXI± inputs.



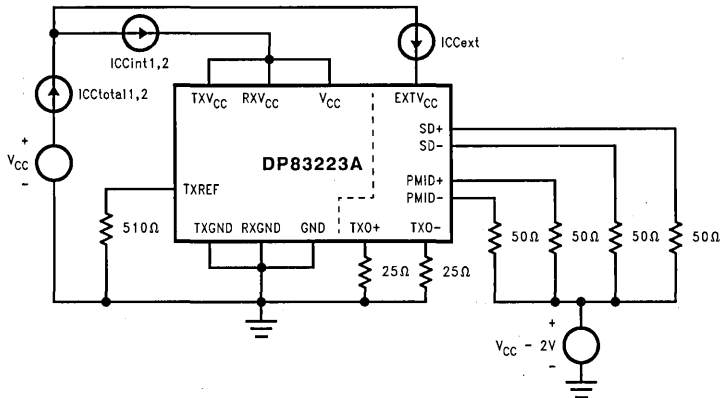
TL/F/11886-40

FIGURE 24a. EQSEL 3 Level Input



TL/F/11886-41

FIGURE 24b. Typical TXO Current vs  $R_{REF}$



TL/F/11886-11

FIGURE 24c.  $I_{CC}$  Diagram



## 5.0 Electrical Characteristics (Continued)

5.3 AC ELECTRICAL CHARACTERISTICS  $T_A = 25^\circ\text{C}$ ,  $V_{CC} = 5.0\text{V}$

| Symbol            | Parameter                            | Conditions                                    | Min | Typ  | Max | Units         |
|-------------------|--------------------------------------|-----------------------------------------------|-----|------|-----|---------------|
| $t_{TX1/f1}$      | TXO Rise/Fall (10%–90%)              | ENCSEL = 1, <i>Figure 25</i>                  |     | 2.2  |     | ns            |
| $t_{TX1/f2}$      | TXO Rise/Fall (10%–90%)              | ENCSEL = 0, <i>Figure 26</i>                  |     | 2.2  |     | ns            |
| $t_{TLH}$         | PMID Rise (20%–80%)                  | LBEN = 1, <i>Figure 27</i>                    |     | 1.0  |     | ns            |
| $t_{THL}$         | PMID Fall (20%–80%)                  | LBEN = 1, <i>Figure 27</i>                    |     | 1.0  |     | ns            |
| $t_{TXpd1}$       | PMRD/TXO Prop Delay                  | ENCSEL = 1, <i>Figure 25</i>                  |     | 6.5  |     | ns            |
| $t_{TXpd2}$       | PMRD/TXO Prop Delay                  | ENCSEL = 0, <i>Figure 26</i>                  |     | 7.0  |     | ns            |
| $t_{RXpd1}$       | RXI/PMID Prop Delay                  | ENCSEL = 1 (Note 1) <i>Figure 27</i>          |     | 4.5  |     | ns            |
| $t_{RXpd2}$       | RXI/PMID Prop Delay                  | ENCSEL = 0 (Note 2) <i>Figure 28</i>          |     | 5.5  |     | ns            |
| $t_{LBpd}$        | PMRD/PMID Prop Delay                 | LBEN = 1, <i>Figure 31</i>                    |     | 2.0  |     | ns            |
| $t_{LBon\_pmid}$  | LBEN on to PMID                      | LBEN = 1, RXI Not Switching, <i>Figure 29</i> |     | 20   |     | ns            |
| $t_{LBoff\_pmid}$ | LBEN off to PMID                     | LBEN = 0, RXI Not Switching, <i>Figure 29</i> |     | 20   |     | ns            |
| $t_{LBon\_txo}$   | LBEN on to TXO Balance               | LBEN = 1, <i>Figure 29</i>                    |     | 50   |     | ns            |
| $t_{LBoff\_txo}$  | LBEN off to TXO Active               | LBEN = 0, <i>Figure 29</i>                    |     | 20   |     | ns            |
| $t_{LBon\_sd}$    | LBEN on to SD+ High                  | LBEN = 1, <i>Figure 29</i>                    |     | 20   |     | ns            |
| $t_{LBoff\_sd}$   | LBEN off to SD+ Normal               | LBEN = 0, <i>Figure 29</i>                    |     | 20   |     | ns            |
| $t_{OEon\_txo}$   | $\overline{OE}$ on to TXO Active     | <i>Figure 30</i>                              |     | 40   |     | ns            |
| $t_{OEoff\_txo}$  | $\overline{OE}$ off to TXO TRI-STATE | <i>Figure 30</i>                              |     | 20   |     | ns            |
| $t_{SDon1}$       | RXI Data to SD High                  | ENCSEL = 1 (Note 3) <i>Figure 32</i>          |     | 28   |     | $\mu\text{s}$ |
| $t_{SDon2}$       | RXI Data to SD High                  | ENCSEL = 0 (Note 4) <i>Figure 33</i>          |     | 0.5  |     | $\mu\text{s}$ |
| $t_{SDoff1}$      | RXI Quiet to SD Low                  | ENCSEL = 1 (Note 3) <i>Figure 32</i>          |     | 28   |     | $\mu\text{s}$ |
| $t_{SDoff2}$      | RXI Quiet to SD Low                  | ENCSEL = 0 (Note 4) <i>Figure 33</i>          |     | 0.5  |     | $\mu\text{s}$ |
| $t_{jTX1}$        | TX pk Total Jitter                   | ENCSEL = 1 (Note 5)                           |     | 750  |     | ps            |
| $t_{jTX2}$        | TX pk Total Jitter                   | ENCSEL = 0 (Note 5)                           |     | 850  |     | ps            |
| $t_{jRX1}$        | RX pk Total Jitter                   | ENCSEL = 1 (Note 6)                           |     | 1400 |     | ps            |
| $t_{jRX2}$        | RX pk Total Jitter                   | ENCSEL = 0 (Note 6)                           |     | 1500 |     | ps            |
| Mbps              | MLT-3 Data Rate                      | ENCSEL = 0                                    |     | 125  |     | Mbps          |

**Note 1:**  $t_{RXpd1}$  RXI to PMID timing is measured by applying a 1 Vp-p 62.5 MHz square wave to the RXI $\pm$  inputs with EQSEL floating.

**Note 2:**  $t_{RXpd2}$  RXI to PMID timing is measured by applying a 1.5 Vp-p 62.5 MHz MLT-3 IDLE to the RXI $\pm$  input with EQSEL floating.

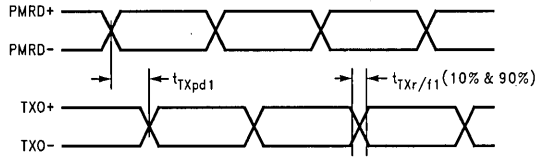
**Note 3:** Signal Detect turn on and turn off times are measured using an MTL-3 2 Vp-p scrambled halt bit stream over a length of Category 5 UTP cable  $\leq$  100 Meters with adaptive equalization enabled.

**Note 4:** Signal Detect turn on and turn off times are measured using a binary 1 Vp-p  $2^{23}$  pseudo random bit stream over a length of Category 5 UTP cable  $\leq$  100 Meters with adaptive equalization enabled.

**Note 5:** TX Jitter measurements are made differentially at the TXO $\pm$  current outputs using a scrambled HALT bit stream for MLT-3 mode and PRBS  $2^{23}$  for binary mode. All measurements are referenced to the original transmit clock.

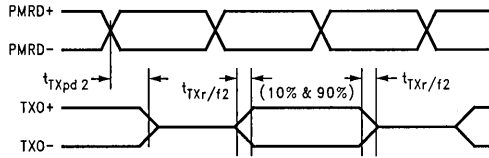
**Note 6:** RX Jitter measurements are made differentially at the PMID $\pm$  PECL outputs using a scrambled HALT bit stream for MLT-3 mode and PRBS  $2^{23}$  for binary mode. This test includes data transmission over a 100M Cat-5 cable plant comprised of 90M of horizontal wiring, two 5M lengths of vertical wiring, two 110 punchdown blocks and 2 RJ45-8 wall jacks. All measurements are referenced to the original transmit clock.

5.0 Electrical Characteristics (Continued)



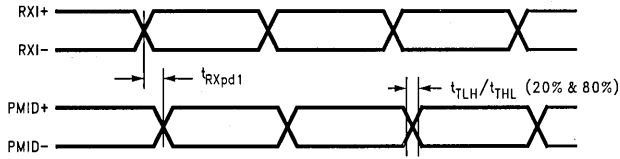
TL/F/11886-42

FIGURE 25. TXO Timing (binary)



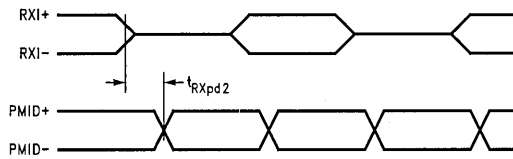
TL/F/11886-43

FIGURE 26. TXO Timing (MLT-3)



TL/F/11886-15

FIGURE 27. PMID Timing (Binary RXI to PMID)



TL/F/11886-44

FIGURE 28. PMID Timing (MLT-3 RXI to PMID)

5.0 Electrical Characteristics (Continued)

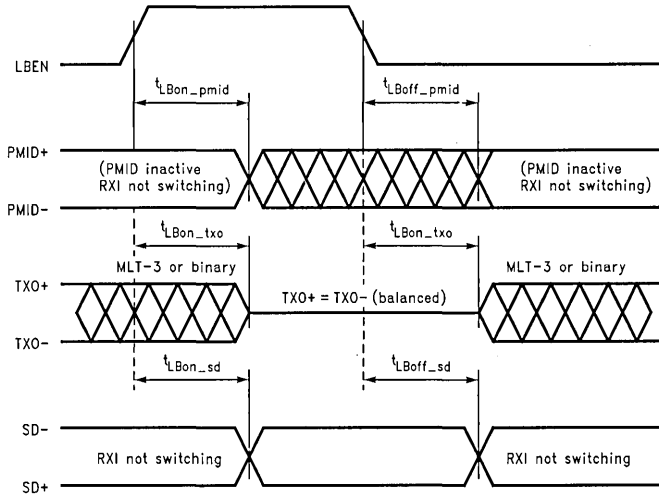


FIGURE 29. Loopback Timing

TL/F/11886-45

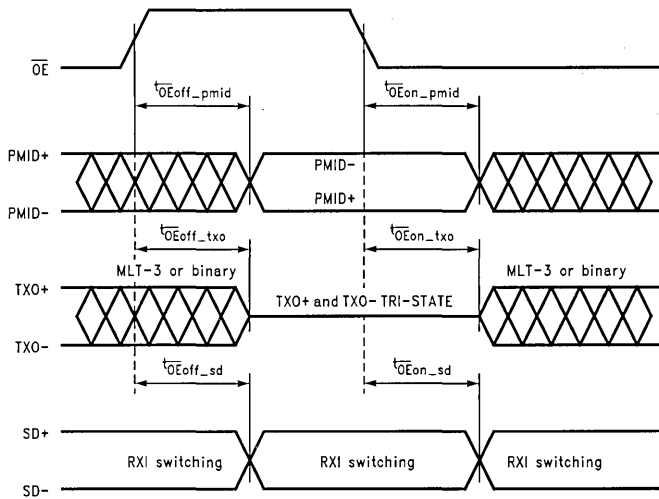


FIGURE 30. Output Enable Timing

TL/F/11886-46

### 5.0 Electrical Characteristics (Continued)

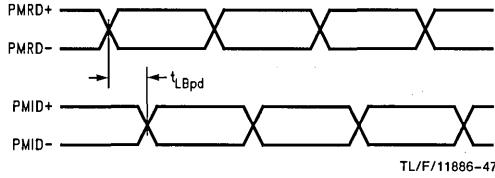


FIGURE 31. PMID Timing (Loopback)

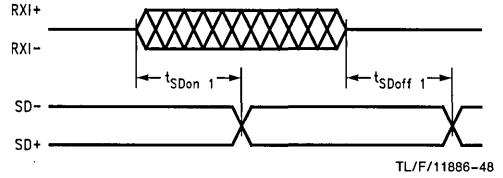


FIGURE 32. Signal Detect Timing (Binary RXI to SD)

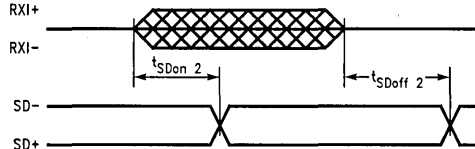
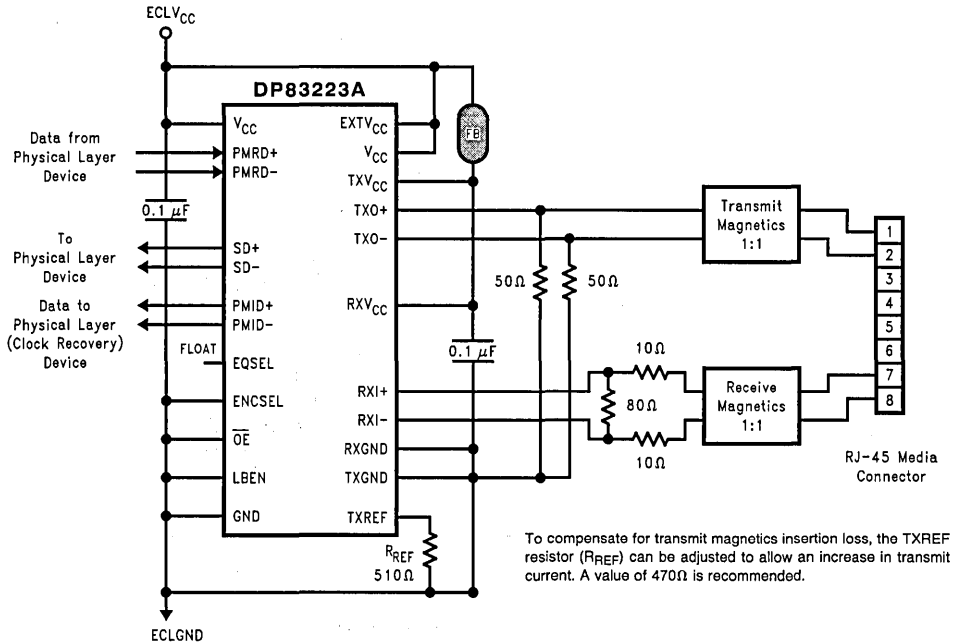


FIGURE 33. Signal Detect Timing (MLT-3 RXI to SD)

TL/F/11886-49



To compensate for transmit magnetics insertion loss, the TXREF resistor ( $R_{REF}$ ) can be adjusted to allow an increase in transmit current. A value of 470Ω is recommended.

= Fair-Rite bead #274-3019-446

The PMRD, SD and PMID 100K ECL differential signal must be terminated into a standard ECL load of 50Ω to  $V_{CC}$  minus 2.0V or equivalent: i.e., Thevenin of 130Ω to GND accompanied by 82Ω to  $V_{CC}$ .

FIGURE 34. Typical Schematic for MTL-3 Signalling over Cat-5 UTP 100Ω Cable

TL/F/11886-16

## DP83840

### 10/100 Mb/s Ethernet Physical Layer

#### General Description

The DP83840 is a Physical Layer device for Ethernet 10BASE-T and 100BASE-X using category 5 Unshielded, Type 1 Shielded and Fiber Optic cables.

This VLSI device is designed for easy implementation of 10/100 Mb/s Ethernet LANs. It interfaces to the PMD sub-layer through National Semiconductor's DP83223 Twisted Pair Transceiver, and to the MAC layer through a Media Independent Interface (MII), ensuring interoperability between products from different vendors.

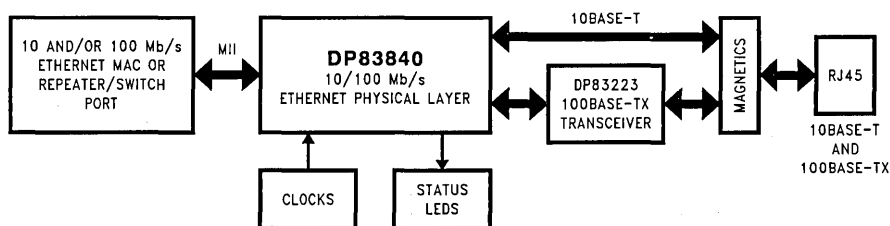
The DP83840 is designed with National Semiconductor's BiCMOS process. Its system architecture is based on the integration of several of National Semiconductor's industry proven core technologies as listed below:

- 10BASE-T ENDEC/Transceiver module to provide the 10 Mb/s IEEE 802.3 functions
- Clock Recovery/Generator Modules from National Semiconductor's leading FDDI product
- FDDI Stream Cipher (Cyclone)
- 100BASE-X physical coding sub-layer (PCS) and control logic that integrate the core modules into a dual speed Ethernet physical layer controller

#### Features

- IEEE 802.3 10BASE-T compatible—ENDEC and UTP/STP transceivers and filters built-in
- IEEE 802.3u 100BASE-X compatible—support for 2 pair Category 5 UTP (100m), Type 1 STP and Fiber Optic Transceivers—Connects directly to the DP83223 Twisted Pair Transceiver
- ANSI X3T12 TP-PMD compatible
- IEEE 802.3u Auto-Negotiation for automatic speed selection
- IEEE 802.3u compatible Media Independent Interface (MII) with Serial Management Interface
- Integrated high performance 100 Mb/s clock recovery circuitry requiring no external filters
- Full Duplex support for 10 and 100 Mb/s
- MII Serial 10 Mb/s output mode
- Fully configurable node and repeater modes—allows operation in either application
- Programmable loopback modes for easy system diagnostics
- Flexible LED support
- IEEE 1149.1 Standard Test Access Port and Boundary-Scan compatible
- Small footprint 100-pin PQFP package

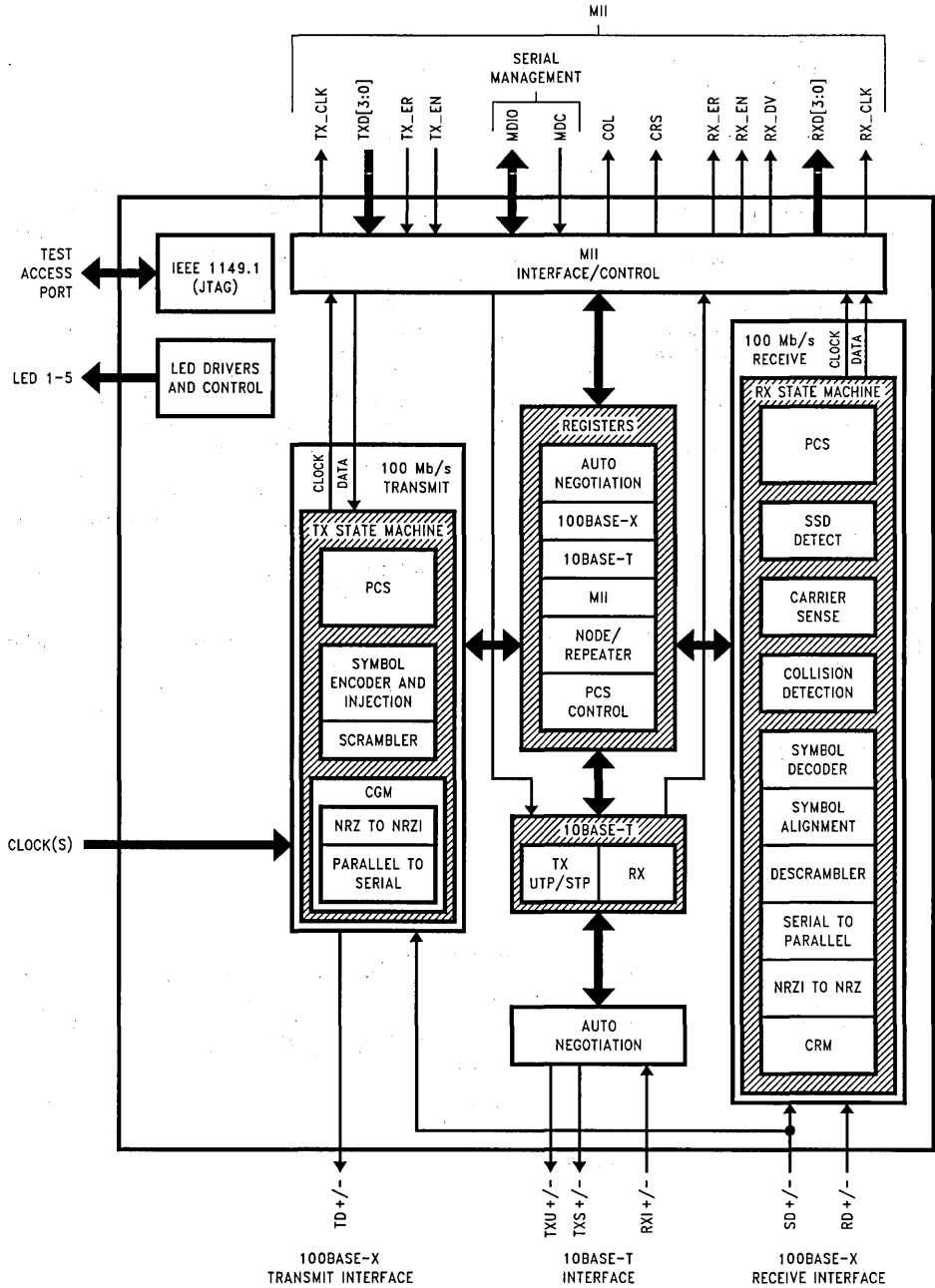
#### System Diagram



TL/F/12388-1

**U.S. Patents Pending**

# Block Diagram



TL/F/12388-2

## Table of Contents

### GENERAL DESCRIPTION

### FEATURES

### SYSTEM DIAGRAM

### BLOCK DIAGRAM

### TABLE OF CONTENTS

#### 1.0 PIN CONNECTION DIAGRAM

#### 2.0 PIN DESCRIPTION

- 2.1 MII Interface
- 2.2 100 Mb/s Serial PMD Interface
- 2.3 10BASE-T Transceiver Module
- 2.4 Clock Interface
- 2.5 Device Configuration Interface
- 2.6 LED Interface
- 2.7 IEEE 1149.1 Interface
- 2.8 PHY Address Interface
- 2.9 Miscellaneous
- 2.10 Power and Ground Pins
- 2.11 Special Connect Pins

#### 3.0 FUNCTIONAL DESCRIPTION

- 3.1 PCS Control
- 3.2 MII Serial Management Register Access
- 3.3 100BASE-X Transmitter
- 3.4 100BASE-X Receiver
- 3.5 Clock Generation Module
- 3.6 100 Mb/s Clock Recovery Module
- 3.7 10BASE-T Transceiver Module
- 3.8 IEEE 1149.1 Controller
- 3.9 IEEE 802.3u Auto-Negotiation
- 3.10 Reset Operation
- 3.11 Loopback Operation
- 3.12 Alternative 100BASE-X Operation

#### 4.0 Registers

- 4.1 Key to Defaults
- 4.2 Basic Mode Control Register
- 4.3 Basic Mode Status Register
- 4.4 PHY Identifier Register # 1
- 4.5 PHY Identifier Register # 2
- 4.6 Auto-Negotiation Advertisement Register
- 4.7 Auto-Negotiation Link Partner Ability Register
- 4.8 Auto-Negotiation Expansion Register
- 4.9 Disconnect Counter Register
- 4.10 False Carrier Sense Counter Register
- 4.11 Receive Error Counter Register
- 4.12 Silicon Revision Register
- 4.13 PCS Sub-Layer Configuration Register
- 4.14 Loopback, Bypass and Receiver Error Mask Register
- 4.15 PHY Address Register
- 4.16 10BASE-T Status Register
- 4.17 10BASE-T Configuration Register

#### 5.0 DP83840 APPLICATION

- 5.1 Typical Board Level Application
- 5.2 Layout Recommendations
- 5.3 Plane Partitioning
- 5.4 Power and Ground Filtering

#### 6.0 DC AND AC SPECIFICATIONS

- 6.1 Ratings and Operating Conditions
- 6.2 DC Specifications
- 6.3 AC Specifications

#### 7.0 PACKAGE DIMENSIONS

# 1.0 Pin Connection Diagram

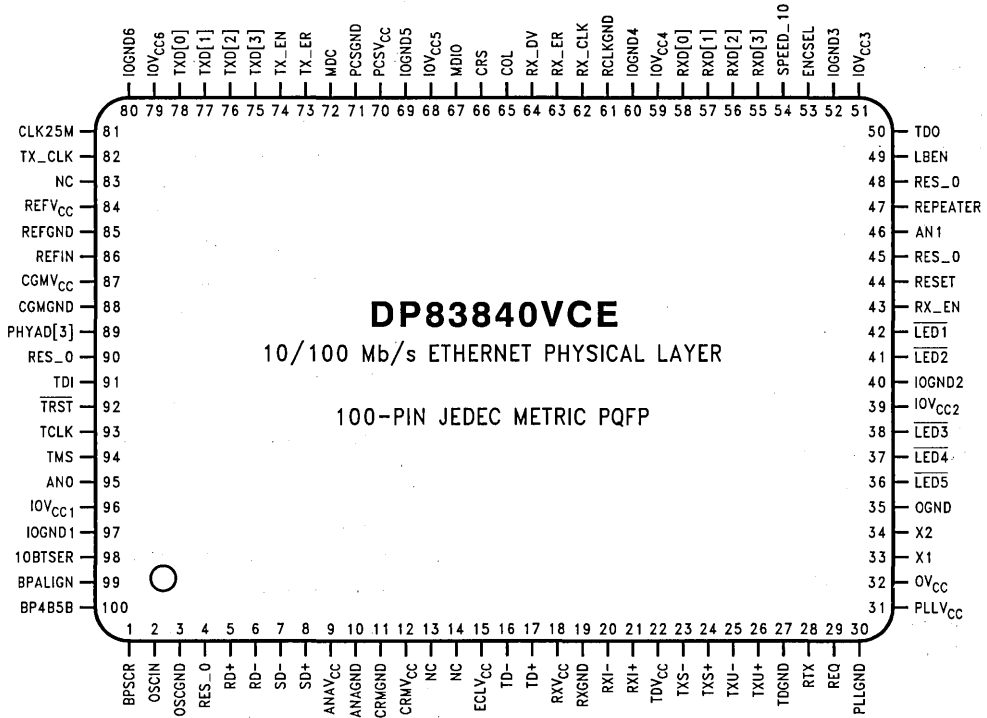


FIGURE 1. DP83840 Pin Connection Diagram

TL/F/12388-3



## 2.0 Pin Description

The DP83840 pins are classified into the following interface categories (each interface is described in the sections that follow):

### MII INTERFACE

100 Mb/s SERIAL PMD INTERFACE

10 Mb/s INTERFACE

CLOCK INTERFACE

DEVICE CONFIGURATION INTERFACE

### LED INTERFACE

IEEE 1149.1 INTERFACE

PHY ADDRESS INTERFACE

MISCELLANEOUS PINS

POWER AND GROUND PINS

SPECIAL CONNECT PINS

### 2.1 MII INTERFACE

| Signal Name                          | Type      | Pin #                | Description                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|--------------------------------------|-----------|----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| TX_CLK                               | O, Z      | 82                   | <p><b>TRANSMIT CLOCK:</b> Transmit clock output from the DP83840:</p> <ul style="list-style-type: none"> <li>— 25 MHz nibble transmit clock derived from Clock Generator Module's (CGM) PLL in 100BASE-TX mode</li> <li>— 2.5 MHz transmit clock in 10BASE-T nibble mode</li> <li>— 10 MHz transmit clock in 10BASE-T serial mode</li> </ul>                                                                                                    |
| TXD[3]<br>TXD[2]<br>TXD[1]<br>TXD[0] | I, J      | 75<br>76<br>77<br>78 | <p><b>TRANSMIT DATA:</b> Transmit data input pins for nibble data from the MII in 100 Mb/s or 10 Mb/s nibble mode (25 MHz for 100 Mb/s mode, 2.5 MHz for 10 Mb/s nibble mode). In 10 Mb/s serial mode, the TXD[0] pin is used as the serial data input pin. TXD[3:1] are ignored.</p>                                                                                                                                                           |
| TX_EN                                | I, J      | 74                   | <p><b>TRANSMIT ENABLE:</b> Active high input indicates the presence of valid nibble data on TXD[3:0] for both 100 Mb/s or 10 Mb/s nibble mode. In 10 Mb/s serial mode, active high indicates the presence of valid 10 Mb/s data on TXD[0].</p>                                                                                                                                                                                                  |
| TX_ER                                | I, J      | 73                   | <p><b>TRANSMIT ERROR:</b> In 100 Mb/s mode, when this signal is high and TX_EN is active the HALT symbol is substituted for the actual data nibble. In 10 Mb/s mode, this input is ignored. In encoder bypass mode (BP_4B5B or BP_ALIGN) TX_ER becomes the TXD[4] pin, the fifth TXD data bit.</p>                                                                                                                                              |
| MDC                                  | I, J      | 72                   | <p><b>MANAGEMENT DATA CLOCK:</b> Synchronous clock to the MDIO management data input/output serial interface which may be asynchronous to transmit and receive clocks. The maximum clock rate is 2.5 MHz.</p>                                                                                                                                                                                                                                   |
| MDIO                                 | I/O, J    | 67                   | <p><b>MANAGEMENT DATA I/O:</b> Bi-directional management instruction/data signal that may be sourced by the station management entity or the PHY. This pin requires a 4.7 k<math>\Omega</math> pullup resistor.</p>                                                                                                                                                                                                                             |
| CRS<br>(PHYAD[2])                    | I/O, Z, J | 66                   | <p><b>CARRIER SENSE:</b> This pin is asserted high to indicate the presence of carrier due to receive or transmit activities in 10BASE-T or 100BASE-X Half Duplex modes. In Repeater, Full Duplex, or Loopback mode a logic 1 indicates presence of carrier due only to receive activity. This is also the PHY address sensing (PHYAD[2]) pin for multiple PHY applications—see Section 2.8 for more details.</p>                               |
| COL                                  | O, Z, J   | 65                   | <p><b>COLLISION DETECT:</b> Asserted high to indicate detection of collision conditions in 10 Mb/s and 100 Mb/s Half Duplex modes. In 10BASE-T Half Duplex mode with Heartbeat asserted (bit 4, register 1Ch), it is also asserted for a duration of approximately 1 <math>\mu</math>s at the end of transmission to indicate CD heartbeat. In Full Duplex mode this signal is always logic 0. There is no heartbeat function in this mode.</p> |
| RX_CLK                               | O, Z      | 62                   | <p><b>RECEIVE CLOCK:</b> Provides the recovered receive clock for different modes of operation:</p> <ul style="list-style-type: none"> <li>— 25 MHz nibble clock in 100 Mb/s mode</li> <li>— 2.5 MHz nibble clock in 10 Mb/s nibble mode</li> <li>— 10 MHz receive clock in 10 Mb/s serial mode</li> </ul>                                                                                                                                      |

I = TTL/CMOS input O = TTL/CMOS output Z = TRI-STATE\* output J = IEEE 1149.1 pin

## 2.0 Pin Description (Continued)

### 2.1 MII INTERFACE (Continued)

| Signal Name                          | Type    | Pin #                | Description                                                                                                                                                                                                                                                                                                                                                                                      |
|--------------------------------------|---------|----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| RX_ER<br>(PHYAD[4])                  | O, Z, J | 63                   | <b>RECEIVE ERROR:</b> Asserted high to indicate that an invalid symbol has been detected inside a received packet in 100 Mb/s mode.<br><br>In a 5B/4B decoder bypass mode (BP_4B5B or BP_ALIGN modes), RX_ER becomes RXD[4], the fifth RXD data bit of the 5B symbol.<br><br>This is also the PHY address sensing (PHYAD[4]) pin for multiple PHY applications—see Section 2.8 for more details. |
| RX_DV                                | O, Z, J | 64                   | <b>RECEIVE DATA VALID:</b> Asserted high to indicate that valid data is present on RXD[3:0].                                                                                                                                                                                                                                                                                                     |
| RXD[3]<br>RXD[2]<br>RXD[1]<br>RXD[0] | O, Z, J | 55<br>56<br>57<br>58 | <b>RECEIVE DATA:</b> Nibble wide receive data (synchronous to RX_CLK—25 MHz for 100BASE-X mode, 2.5 MHz for 10BASE-T nibble mode). Data is driven on the falling edge of RX_CLK.<br><br>In 10 Mb/s serial mode, the RXD[0] pin is used as the data output pin. RXD[3:1] are don't care.                                                                                                          |
| RX_EN                                | I, J    | 43                   | <b>RECEIVE ENABLE:</b> Active high enable for receive signals RXD[3:0], RX_CLK, RX_DV and RX_ER. A low on this input tri-states these output pins. For normal operation in a node application this pin should be pulled high.                                                                                                                                                                    |

I = TTL/CMOS input O = TTL/CMOS output Z = TRI-STATE output J = IEEE 1149.1 pin

### 2.2 100 Mb/S SERIAL PMD INTERFACE

| Signal Name          | Type    | Pin #    | Description                                                                                                                                                                                                                                                                                                                                             |
|----------------------|---------|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| SPEED_10             | O, J    | 54       | <b>SPEED 10 Mb/s:</b> Indicates 10 Mb/s operation when high. Indicates 100 Mb/s operation when low. This pin can be used to drive a low current LED to indicate 100 Mb/s speed if required.                                                                                                                                                             |
| ENCSEL<br>(PHYAD[1]) | I/O, J  | 53       | <b>ENCODE SELECT:</b> Used to select binary or MLT-3 coding scheme in the PMD transceiver (at the DP83223, logic high selects binary coding scheme and logic low selects MLT-3 coding scheme).<br><br>This is also the PHY address sensing (PHYAD[1]) pin for multiple PHY applications—see Section 2.8 for more details.                               |
| LBEN<br>(PHYAD[0])   | I/O, J  | 49       | <b>LOOPBACK ENABLE:</b> This pin should be connected to the Loopback Enable pin of a DP83223 100 Mb/s Transceiver:<br><br>1 = Loopback enabled<br>0 = Loopback disabled<br><br>In 10 Mb/s modes, this output has no meaning.<br><br>This is also the PHY address sensing (PHYAD[0]) pin for multiple PHY applications—see Section 2.8 for more details. |
| TD–<br>TD+           | O (ECL) | 16<br>17 | <b>TRANSMIT DATA:</b> Differential ECL 125 Mb/s serialized transmit data outputs to the DP83223 Twister.                                                                                                                                                                                                                                                |
| SD–<br>SD+           | I (ECL) | 7<br>8   | <b>SIGNAL DETECT:</b> Differential ECL signal detect inputs. Indicates that a signal is present at the DP83223 receive inputs as specified by the TP-PMD ANSI standard.                                                                                                                                                                                 |
| RD–<br>RD+           | I (ECL) | 6<br>5   | <b>RECEIVE DATA:</b> Differential ECL 125 Mb/s receive data inputs.                                                                                                                                                                                                                                                                                     |

I = TTL/CMOS input O = TTL/CMOS output Z = TRI-STATE output J = IEEE 1149.1 pin

## 2.0 Pin Description (Continued)

### 2.3 10 Mb/S INTERFACE

| Signal Name  | Type | Pin #    | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|--------------|------|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| REQ          | I    | 29       | <b>EQUALIZATION RESISTOR:</b> A resistor connected between this pin and GND or $V_{CC}$ adjusts the equalization step amplitude on the 10BASE-T Manchester encoded transmit data (TXU+/- or TXS+/-). Typically no resistor is required for operation with cable lengths less than 100m. Great care must be taken to ensure system timing integrity when using cable lengths greater than 100m. Refer to the IEEE 802.3u standard, Clause 29 for more details on system topology issues.<br><br>The equations to calculate this resistor value are still under investigation. Currently, this value must be determined empirically. |
| RTX          | I    | 28       | <b>EXTENDED CABLE RESISTOR:</b> A resistor connected between this pin and GND or $V_{CC}$ adjusts the amplitude of the differential transmit outputs (TXU+/- or TXS+/-). Typically no resistor is required for operation with cable lengths less than 100m. Great care must be taken to ensure system timing integrity when using cable lengths greater than 100m. Refer to the IEEE 802.3u standard, Clause 29 for more details on system topology issues.<br><br>The equations to calculate this resistor value are still under investigation. Currently, this value must be determined empirically.                             |
| TXU-<br>TXU+ | O    | 25<br>26 | <b>UNSHIELDED TWISTED PAIR OUTPUT:</b> This differential output pair is the filtered 10BASE-T transmit data for UTP cable.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| TXS-<br>TXS+ | O    | 23<br>24 | <b>SHIELDED TWISTED PAIR OUTPUT:</b> This differential output pair is the filtered 10BASE-T transmit data for STP cable.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| RXI-<br>RXI+ | I    | 20<br>21 | <b>TWISTED PAIR RECEIVE INPUT:</b> These are the differential 10BASE-T receive data inputs for either STP or UTP.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

I = TTL/CMOS input O = TTL/CMOS output Z = TRI-STATE output J = IEEE 1149.1 pin

### 2.4 CLOCK INTERFACE

| Signal Name | Type | Pin # | Description                                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| REFIN       | I    | 86    | <b>REFERENCE INPUT:</b> 25 MHz TTL reference clock input. Can be supplied from an external oscillator module or from the CLK25M output.                                                                                                                                                                                                                                                                                            |
| CLK25M      | O, Z | 81    | <b>25 MHz CLOCK OUTPUT:</b> Derived from the 50 MHz OSCIN input.                                                                                                                                                                                                                                                                                                                                                                   |
| OSCIN       | I    | 2     | <b>OSCILLATOR INPUT:</b> 50 MHz $\pm$ 50 ppm external TTL oscillator input. If not used, pull down to GND with a 4.7 k $\Omega$ resistor.                                                                                                                                                                                                                                                                                          |
| X2          | O    | 34    | <b>CRYSTAL OSCILLATOR OUTPUT:</b> External 20 MHz $\pm$ 0.005% crystal connection. Used for 10BASE-T timing. When using an external 20 MHz oscillator connected to X1, leave this pin unconnected.                                                                                                                                                                                                                                 |
| X1          | I    | 33    | <b>CRYSTAL OSCILLATOR INPUT:</b> External 20 MHz $\pm$ 0.005% crystal connection. Used for 10BASE-T timing and Auto-Negotiation. If not used, this pin should be tied to $V_{CC}$ either directly or via a pull-up resistor—typically 4.7 k $\Omega$ . The DP83840 detects this condition, enables the internal $\div$ 2.5 divider and switches the 10 Mb/s and Auto-Negotiation circuitry to the internally derived 20 MHz clock. |

I = TTL/CMOS input O = TTL/CMOS output Z = TRI-STATE output J = IEEE 1149.1 pin

## 2.0 Pin Description (Continued)

### 2.5 DEVICE CONFIGURATION INTERFACE

| Signal Name | Type | Pin #                                                                                      | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |
|-------------|------|--------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-----|-------------|---|---|------------------------------------------------|---|---|------------------------------------------------|---|---|--------------------------------------------------|---|---|--------------------------------------------------|-----|-----|-----------------|---|---|--------------------------------------------------------------------------------------------|---|---|-------------------------------------------------------|---|---|-------------------------------------------------------|---|---|---------------------------------------------------------|---|---|---------------------------------------------------------|
| AN0         | I    | 95                                                                                         | <p><b>AN0:</b> This is a three level input pin (i.e., 1, M, 0) that works in conjunction with the AN1 pin to control the forced or advertised operating mode of the DP83840 according to the following table. The value on this pin is set by either connecting the input to GND or <math>V_{CC}</math> (0 or 1) or leaving it unconnected (M). The unconnected state, M, refers to the mid level (<math>V_{CC} \div 2</math>) set by internal resistors (<math>\sim 3 \text{ k}\Omega</math>). This value is latched into the DP83840 at power-up/reset. See Section 3.9 for more details.</p> <table border="0"> <thead> <tr> <th>AN1</th> <th>AN0</th> <th>Forced Mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>M</td> <td>10BASE-T, Half-Duplex without Auto-Negotiation</td> </tr> <tr> <td>1</td> <td>M</td> <td>10BASE-T, Full Duplex without Auto-Negotiation</td> </tr> <tr> <td>M</td> <td>0</td> <td>100BASE-TX, Half-Duplex without Auto-Negotiation</td> </tr> <tr> <td>M</td> <td>1</td> <td>100BASE-TX, Full Duplex without Auto-Negotiation</td> </tr> </tbody> </table><br><table border="0"> <thead> <tr> <th>AN1</th> <th>AN0</th> <th>Advertised Mode</th> </tr> </thead> <tbody> <tr> <td>M</td> <td>M</td> <td>All capable (i.e. Full Duplex for 10BASE-T and 100BASE-TX) advertised via Auto-Negotiation</td> </tr> <tr> <td>0</td> <td>0</td> <td>10BASE-T, Half-Duplex advertised via Auto-Negotiation</td> </tr> <tr> <td>0</td> <td>1</td> <td>10BASE-T, Full Duplex advertised via Auto-Negotiation</td> </tr> <tr> <td>1</td> <td>0</td> <td>100BASE-TX, Half-Duplex advertised via Auto-Negotiation</td> </tr> <tr> <td>1</td> <td>1</td> <td>100BASE-TX, Full Duplex advertised via Auto-Negotiation</td> </tr> </tbody> </table> | AN1 | AN0 | Forced Mode | 0 | M | 10BASE-T, Half-Duplex without Auto-Negotiation | 1 | M | 10BASE-T, Full Duplex without Auto-Negotiation | M | 0 | 100BASE-TX, Half-Duplex without Auto-Negotiation | M | 1 | 100BASE-TX, Full Duplex without Auto-Negotiation | AN1 | AN0 | Advertised Mode | M | M | All capable (i.e. Full Duplex for 10BASE-T and 100BASE-TX) advertised via Auto-Negotiation | 0 | 0 | 10BASE-T, Half-Duplex advertised via Auto-Negotiation | 0 | 1 | 10BASE-T, Full Duplex advertised via Auto-Negotiation | 1 | 0 | 100BASE-TX, Half-Duplex advertised via Auto-Negotiation | 1 | 1 | 100BASE-TX, Full Duplex advertised via Auto-Negotiation |
| AN1         | AN0  | Forced Mode                                                                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |
| 0           | M    | 10BASE-T, Half-Duplex without Auto-Negotiation                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |
| 1           | M    | 10BASE-T, Full Duplex without Auto-Negotiation                                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |
| M           | 0    | 100BASE-TX, Half-Duplex without Auto-Negotiation                                           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |
| M           | 1    | 100BASE-TX, Full Duplex without Auto-Negotiation                                           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |
| AN1         | AN0  | Advertised Mode                                                                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |
| M           | M    | All capable (i.e. Full Duplex for 10BASE-T and 100BASE-TX) advertised via Auto-Negotiation |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |
| 0           | 0    | 10BASE-T, Half-Duplex advertised via Auto-Negotiation                                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |
| 0           | 1    | 10BASE-T, Full Duplex advertised via Auto-Negotiation                                      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |
| 1           | 0    | 100BASE-TX, Half-Duplex advertised via Auto-Negotiation                                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |
| 1           | 1    | 100BASE-TX, Full Duplex advertised via Auto-Negotiation                                    |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |
| AN1         | I    | 46                                                                                         | <p><b>AN1:</b> This is a three level input pin (i.e., 1, M, 0) that works in conjunction with the AN0 pin to control the forced or advertised operating mode of the DP83840 according to the table given in the AN0 pin description above. The value on this pin is set by either connecting the input to GND or <math>V_{CC}</math> (0 or 1) or leaving it unconnected (M). This value is latched into the DP83840 at power-up/reset. See Section 3.9 for more details.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |
| REPEATER    | I, J | 47                                                                                         | <p><b>REPEATER/NODE MODE:</b> Selects REPEATER mode when set high and NODE mode when set low. In REPEATER mode or NODE mode with Full Duplex configured, the Carrier Sense (CRS) output from the DP83840 is asserted due to receive activity only. In NODE mode, and not configured for Full Duplex operation, CRS is asserted due to either receive and transmit activity.</p> <p>At power-up/reset, the value on this pin (set by a pull-up or pull-down resistor, typically <math>4.7 \text{ k}\Omega</math>) is latched to bit 12 of the PCS Configuration Register, address 17h.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |
| 10BTSER     | I, J | 98                                                                                         | <p><b>SERIAL/NIBBLE SELECT:</b></p> <p>10 Mb/s Serial Operation:</p> <p>When set high, this input selects serial data transfer mode. Manchester encoded transmit and receive data is exchanged serially with a 10 MHz clock rate on the least significant bits of the nibble-wide MII data buses, pins TXD[0] and RXD[0] respectively. This mode is intended for use with the DP83840 connected to a device (MAC or Repeater) that has a 10 Mb/s serial interface. Serial operation is not supported in 100 Mb/s mode, so for 100 Mb/s this input is ignored.</p> <p>10 and 100 Mb/s Nibble Operation:</p> <p>When set low, this input selects the MII compliant nibble data transfer mode. Transmit and receive data is exchanged in nibbles on the TXD[3:0] and RXD[3:0] pins respectively.</p> <p>At power-up/reset, the value on this pin (set by a pull-up or pull-down resistor, typically <math>4.7 \text{ k}\Omega</math>) is latched to bit 9 of the 10BASE-T Status Register, address 1Bh.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |
| BPALIGN     | I, J | 99                                                                                         | <p><b>BYPASS ALIGNMENT:</b> Allows 100 Mb/s transmit and receive data streams to bypass all of the transmit and receive operations when set high. Refer to <i>Figures 4 and 5</i>.</p> <p>At power-up/reset, the value on this pin (set by a pull-up or pull-down resistor, typically <math>4.7 \text{ k}\Omega</math>) is latched into bit 12 of the Loopback, Bypass and Receiver Error Mask Register, address 18h.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |     |     |             |   |   |                                                |   |   |                                                |   |   |                                                  |   |   |                                                  |     |     |                 |   |   |                                                                                            |   |   |                                                       |   |   |                                                       |   |   |                                                         |   |   |                                                         |

I = TTL/CMOS input O = TTL/CMOS output Z = TRI-STATE output J = IEEE 1149.1 pin

## 2.0 Pin Description (Continued)

### 2.5 DEVICE CONFIGURATION INTERFACE (Continued)

| Signal Name           | Type | Pin # | Description                                                                                                                                                                                                                                                                                                                                                                     |
|-----------------------|------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| B $\overline{2}$ 4B5B | I, J | 100   | <b>BYPASS 4B5B ENCODER/DECODER:</b> Allows 100 Mb/s transmit and receive data streams to bypass the 4B to 5B encoder and 5B to 4B decoder circuits when set high.<br>At power-up/reset, the value on this pin (set by a pull-up or pull-down resistor, typically 4.7 k $\Omega$ ) is latched into bit 14 of the Loopback, Bypass and Receiver Error Mask Register, address 18h. |
| BPSCR                 | I, J | 1     | <b>BYPASS SCRAMBLER/DESCRAMBLER:</b> Allows 100 Mb/s transmit and receive data streams to bypass the scrambler and descrambler circuits when set high.<br>At power-up/reset, the value on this pin (set by a pull-up or pull-down resistor, typically 4.7 k $\Omega$ ) is latched into bit 13 of the Loopback, Bypass and Receiver Error Mask Register, address 18h.            |

I = TTL/CMOS input O = TTL/CMOS output Z = TRI-STATE output J = IEEE 1149.1 pin

### 2.6 LED INTERFACE

These outputs can be used to drive LEDs directly, or can be used to provide status information to a network management device. Refer to *Figure 12* for the LED connection diagram. An LED indication of 100 Mb/s operation can be obtained by connecting a low current LED (and its associated resistor) to the SPEED $\overline{10}$  pin (54). See Section 2.2.

| Signal Name              | Type | Pin # | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|--------------------------|------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| $\overline{\text{LED1}}$ | O, J | 42    | <b>TRANSMIT LED:</b> Indicates the presence of transmit activity for 10 Mb/s and 100 Mb/s operation. Active low.<br>If bit 2 (LED1 $\overline{\text{MODE}}$ ) of the PCS Configuration Register (address 17h) is set high, then the $\overline{\text{LED1}}$ pin function is changed to indicate the status of the Disconnect Function as defined by the state of bit 5 (CON $\overline{\text{STATUS}}$ ) in the PHY address register (address 19h).<br>The DP83840 incorporates a "monostable" function on the $\overline{\text{LED1}}$ output. This ensures that even minimum size packets generate adequate LED ON time to be visible. |
| $\overline{\text{LED2}}$ | O, J | 41    | <b>RECEIVE LED:</b> Indicates the presence of any receive activity (CRS active) for 10 Mb/s and 100 Mb/s operation. Active low.<br>The DP83840 incorporates a "monostable" function on the $\overline{\text{LED2}}$ output. This ensures that even minimum size packets generate adequate LED ON time to be visible.                                                                                                                                                                                                                                                                                                                      |
| $\overline{\text{LED3}}$ | O, J | 38    | <b>LINK LED:</b> Indicates Good Link status for 10 Mb/s and 100 Mb/s operation. Active low.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| $\overline{\text{LED4}}$ | O, J | 37    | <b>POLARITY/FULL DUPLEX LED:</b> Indicates Good Polarity status for 10 Mb/s operation. Indicates Full Duplex mode status for 100 Mb/s operation. Active low.<br>If bit 1 (LED4 $\overline{\text{MODE}}$ ) in the PCS Configuration Register (address 17h) is set high, the $\overline{\text{LED4}}$ pin function is changed to indicate Full Duplex mode status for 10 Mb/s and 100 Mb/s operation.                                                                                                                                                                                                                                       |
| $\overline{\text{LED5}}$ | O, J | 36    | <b>COLLISION LED:</b> Indicates the presence of collision activity for 10 Mb/s and 100 Mb/s operation. This LED has no meaning for 10 Mb/s or 100 Mb/s Full Duplex operation. Active low.                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

I = TTL/CMOS input O = TTL/CMOS output Z = TRI-STATE output J = IEEE 1149.1 pin

### 2.7 IEEE 1149.1 INTERFACE

The IEEE 1149.1 Standard Test Access Port and Boundary Scan (sometimes referred to as JTAG) interface signals allow system level boundary scan to be performed.

| Signal Name              | Type | Pin # | Description                                                                                                                                                                                                                                               |
|--------------------------|------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| TDO                      | O, Z | 50    | <b>TEST DATA OUTPUT:</b> Serial instruction/test output data for the IEEE 1149.1 scan chain.<br>If Boundary-Scan is not implemented this pin should be left unconnected (NC).                                                                             |
| TDI                      | I    | 91    | <b>TEST DATA INPUT:</b> Serial instruction/test input data for the IEEE 1149.1 scan chain.                                                                                                                                                                |
| $\overline{\text{TRST}}$ | I    | 92    | <b>TEST RESET:</b> An asynchronous low going pulse will reset and initialize the IEEE 1149.1 test circuitry.<br>If Boundary-Scan is not implemented, this pin should be left unconnected (NC) since it has an internal pull-up resistor (10 k $\Omega$ ). |

I = TTL/CMOS input O = TTL/CMOS output Z = TRI-STATE output J = IEEE 1149.1 pin

## 2.0 Pin Description (Continued)

### 2.7 IEEE 1149.1 INTERFACE (Continued)

| Signal Name | Type | Pin # | Description                                                                                                                                                                                                            |
|-------------|------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| TCLK        | I    | 93    | <b>TEST CLOCK:</b> Test clock for the IEEE 1149.1 circuitry.<br>This pin should be pulled to GND with an appropriate resistor (10 k $\Omega$ ).                                                                        |
| TMS         | I    | 94    | <b>TEST MODE SELECT:</b> Control input to the IEEE 1149.1 test circuitry.<br>If Boundary-Scan is not implemented, this pin should be left unconnected (NC) since it has an internal pull-up resistor (10 k $\Omega$ ). |

I = TTL/CMOS input O = TTL/CMOS output Z = TRI-STATE output J = IEEE 1149.1 pin

### 2.8 PHY ADDRESS INTERFACE

It should be noted that while PHYAD[4:0] provides up to 32 unique PHY address options, an address selection of all zeros (00000) will result in a PHY isolation condition. See the Isolate bit description in the BMCR, address 00h, Section 4.2.

| Signal Name          | Type      | Pin # | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------|-----------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LBEN<br>(PHYAD[0])   | I/O, J    | 49    | <b>PHY ADDRESS [0]:</b> PHY address sensing pin (bit 0) for multiple PHY applications. PHY address sensing is achieved by strapping a pull-up/pull-down resistor (typically 10 k $\Omega$ ) to this pin as required.<br>The pull-up/pull-down status of this pin is latched into the PHYAD address register (address 19h) during power up/reset.<br>This pin is also the Loopback Enable output pin (LBEN) for the 100 Mb/s Serial PMD Interface. See Section 2.2 for more details. |
| ENCSEL<br>(PHYAD[1]) | I/O, J    | 53    | <b>PHY ADDRESS [1]:</b> PHY address sensing pin (bit 1) for multiple PHY applications. PHY address sensing is achieved by strapping a pull-up/pull-down resistor (typically 10 k $\Omega$ ) to this pin as required.<br>The pull-up/pull-down status of this pin is latched into the PHYAD address register (address 19h) during power up/reset.<br>This pin is also the Encode Select output pin (ENCSEL) for the 100 Mb/s Serial PMD Interface. See Section 2.2 for more details. |
| CRS<br>(PHYAD[2])    | I/O, J    | 66    | <b>PHY ADDRESS [2]:</b> PHY address sensing pin (bit 2) for multiple PHY applications. PHY address sensing is achieved by strapping a pull-up/pull-down resistor (typically 10 k $\Omega$ ) to this pin as required.<br>The pull-up/pull-down status of this pin is latched into the PHYAD address register (address 19h) during power up/reset.<br>This pin is also the Carrier Sense output pin (CRS) for the MII Interface. See Section 2.1 for more details.                    |
| PHYAD[3]             | I         | 89    | <b>PHY ADDRESS [3]:</b> PHY address sensing pin (bit 3) for multiple PHY applications. PHY address sensing is achieved by strapping a pull-up/pull-down resistor (typically 10 k $\Omega$ ) to this pin as required.<br>The pull-up/pull-down status of this pin is latched into the PHYAD address register (address 19h) during power up/reset.<br>Since this input does not have a dual function, it is a good choice for providing a non-zero PHY address to the DP83840.        |
| RX_ER<br>(PHYAD[4])  | I/O, Z, J | 63    | <b>PHY ADDRESS [4]:</b> PHY address sensing pin (bit 4) for multiple PHY applications. PHY address sensing is achieved by strapping a pull-up/pull-down resistor (typically 10 k $\Omega$ ) to this pin as required.<br>The pull-up/pull-down status of this pin is latched into the PHYAD address register (address 19h) during power up/reset.<br>This pin is also the Receive Error output pin (RX_ER) for the MII Interface. See Section 2.1 for more details.                  |

I = TTL/CMOS input O = TTL/CMOS output Z = TRI-STATE output J = IEEE 1149.1 pin

## 2.0 Pin Description (Continued)

### 2.9 MISCELLANEOUS

| Signal Name | Type | Pin # | Description                                            |
|-------------|------|-------|--------------------------------------------------------|
| RESET       | I, J | 44    | RESET: Active high input that initializes the DP83840. |

I = TTL/CMOS input O = TTL/CMOS output Z = TRI-STATE output J = IEEE 1149.1 pin

### 2.10 POWER AND GROUND PINS

The power ( $V_{CC}$ ) and ground (GND) pins of the DP83840 are grouped in pairs into four categories—TTL/CMOS Input pairs, TTL/CMOS Output and I/O pairs, 10 Mb/s pairs and 100 Mb/s pairs. Great care must be taken with the layout of the power and ground supplies to this device. Each of the four categories of pairs should have its own isolated supplies. More details of the power and ground layout requirements are given in Sections 5.2, 5.3 and 5.4.

| Pin Names                                           | Pin #  | Description                                         |
|-----------------------------------------------------|--------|-----------------------------------------------------|
| <b>GROUP A—TTL/CMOS INPUT SUPPLY PAIRS</b>          |        |                                                     |
| IOV <sub>CC1</sub> , IOGND1                         | 96, 97 | TTL Input/Output Supply # 1                         |
| IOV <sub>CC2</sub> , IOGND2                         | 39, 40 | TTL Input/Output Supply # 2                         |
| IOV <sub>CC3</sub> , IOGND3                         | 51, 52 | TTL Input/Output Supply # 3                         |
| PCSV <sub>CC</sub> , PCSGND                         | 70, 71 | Physical Coding Sublayer Supply                     |
| <b>GROUP B—TTL/CMOS OUTPUT AND I/O SUPPLY PAIRS</b> |        |                                                     |
| IOV <sub>CC4</sub> , IOGND4                         | 59, 60 | TTL Input/Output Supply # 4                         |
| RCLKGND                                             | 61     | Receive Clock Ground, No Paired $V_{CC}$            |
| IOV <sub>CC5</sub> , IOGND5                         | 68, 69 | TTL Input/Output Supply # 5                         |
| IOV <sub>CC6</sub> , IOGND6                         | 79, 80 | TTL Input/Output Supply # 6                         |
| REFV <sub>CC</sub> , REFGND                         | 84, 85 | 25 MHz Clock Supply                                 |
| <b>GROUP C—10 Mb/s SUPPLY PAIRS</b>                 |        |                                                     |
| RXV <sub>CC</sub> , RXGND                           | 18, 19 | Receive Section Supply                              |
| TDV <sub>CC</sub> , TDGND                           | 22, 27 | Transmit Section Supply                             |
| PLL <sub>CC</sub> , PLLGND                          | 31, 30 | Phase Locked Loop Supply                            |
| OV <sub>CC</sub> , OGND                             | 32, 35 | Internal Oscillator Supply                          |
| <b>GROUP D—100 Mb/s SUPPLY PAIRS</b>                |        |                                                     |
| OSCGND                                              | 3      | External Oscillator Input Ground—No Paired $V_{CC}$ |
| ANAV <sub>CC</sub> , ANAGND                         | 9, 10  | Analog Section Supply                               |
| CRMV <sub>CC</sub> , CRMGND                         | 12, 11 | Clock Recovery Module Supply                        |
| ECLV <sub>CC</sub>                                  | 15     | ECL Outputs Supply                                  |
| CGMV <sub>CC</sub> , CGMGND                         | 87, 88 | Clock Generator Module Supply                       |

I = TTL/CMOS input O = TTL/CMOS output Z = TRI-STATE output J = IEEE 1149.1 pin

### 2.11 SPECIAL CONNECT PINS

| Signal Name | Type | Pin #          | Description                                                                                                                                                                         |
|-------------|------|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| NC          |      | 13<br>14<br>83 | <b>NO CONNECT:</b> These pins are reserved for future use. Leave them unconnected (floating).                                                                                       |
| RES__0      |      | 4<br>45        | <b>RESERVED__0:</b> These pins are reserved for future use. Connect them to the nearest ground plane. For future upgradability, connect these pins to GND via 0 $\Omega$ resistors. |
| RES__0      | J    | 48<br>90       | <b>RESERVED__0:</b> These pins are reserved for future use. Connect them to the nearest ground plane. For future upgradability, connect these pins to GND via 0 $\Omega$ resistors. |

I = TTL/CMOS input O = TTL/CMOS output Z = TRI-STATE output J = IEEE 1149.1 pin

## 3.0 Functional Description

The DP83840 10/100 Mb/s Ethernet Physical Layer integrates a 100BASE-T Physical Coding Sub-layer (PCS) and a complete 10BASE-T module in a single chip. It provides a standard Media Independent Interface (MII) to communicate between the Physical Signaling and the Medium Access Control (MAC) layers for both 100BASE-X and 10BASE-T operations. It interfaces to a 100 Mb/s Physical Medium Dependent (PMD) transceiver, such as the DP83223.

The 100BASE-X section of the device consists of the following functional blocks:

- Transmitter
- Receiver
- Clock Generation Module (CGM)
- Clock Recovery Module (CRM)

The 10BASE-T section of the device consists of the 10 Mb/s transceiver module with filters and an ENDEC module.

The 100BASE-X and 10BASE-T sections share the following functional blocks:

- PCS Control
- MII Registers
- IEEE 1149.1 Controller
- IEEE 802.3u Auto-Negotiation

Each of these functional blocks is described below.

### 3.1 PCS CONTROL

The IEEE 802.3u 100BASE-X Standard defines the Physical Coding Sublayer (PCS) as the transmit, receive and carrier sense functions. These functions within the DP83840 are controlled via external pins and internal registers via the MII serial management interface.

#### 3.1.1 100BASE-X Bypass Options

The DP83840 incorporates a highly flexible transmit and receive channel architecture. Each of the major 100BASE-X transmit and receive functional blocks of the DP83840 are selectively bypassable to provide increased flexibility for various applications.

##### 3.1.1.1 Bypass 4B5B and 5B4B

The 100BASE-X 4B5B symbol encoder in the transmit channel and the 100BASE-X 5B4B symbol decoder in the receive channel may be bypassed by setting the BP\_4B5B bit in the LBREMR (bit 14, register address 18h). The default value for this bit is set by the BP4B5B pin 100 at power-up/reset.

##### 3.1.1.2 Bypass Scrambler and Descrambler

The 100BASE-X scrambler in the transmit channel and the 100BASE-X descrambler in the receive channel may be bypassed by setting the BP\_\_SCR bit in the LBREMR (bit 13, register address 18h). The default value for this bit is set by the BPSCR signal (pin 1) at power-up/reset.

##### 3.1.1.3 Bypass NRZI Encoder and Decoder

The 100BASE-X NRZI encoder in the transmit channel and the 100BASE-X NRZI decoder in the receive channel may be bypassed by setting the NRZI\_EN bit in the PCR (bit 15, register address 17h). The default for this bit is a 1, which enables the NRZI encoder and decoder.

##### 3.1.1.4 Bypass Align

The 100BASE-X transmit channel operations (4B5B symbol encoder, scrambler and NRZ) and the 100BASE-X

receive channel operations (NRZI to NRZ, descrambler and 4B5B symbol decoding) may all be bypassed by setting the BP\_ALIGN bit in the LBREMR (bit 12, register address 18h). The default value for this bit is set by the BP\_ALIGN signal (pin 99) at power-up/reset.

The bypass align function is intended for those repeater applications where none of the transmit and receive channel operations are required.

#### 3.1.2 Repeater Mode

The DP83840 Carrier Sense (CRS) operation depends on the value of the REPEATER bit in the PCR (bit 12, register address 17h). When set high, the CRS output (pin 66) is asserted for receive activity only. When set low, the CRS output is asserted for either receive or transmit activity.

The default value for this bit is set by the REPEATER pin 66 at power-up/reset.

#### 3.1.3 MII Control

The DP83840 has 3 basic MII operating modes:

##### 3.1.3.1 100 Mb/s Operation

For 100 Mb/s operation, the MII operates in nibble mode with a clock rate of 25 MHz. This clock rate is independent of bypass conditions.

In normal (non-bypassed) operation the MII data at RXD[3:0] and TXD[3:0] is nibble wide. In bypass mode (BP\_4B5B or BP\_ALIGN set) the MII data takes the form of 5-bit symbols. The lowest significant 4 bits appear on TXD[3:0] and RXD[3:0] as normal, and the most significant bits (TXD[4] and RXD[4]) appear on the TX\_ER and RX\_ER pins respectively.

##### 3.1.3.2 10 Mb/s Nibble Mode Operation

For 10 Mb/s nibble mode operation, the MII clock rate is 2.5 MHz. The 100BASE-X bypass functions do not apply to 10 Mb/s operation. This is the default 10 Mb/s mode of operation.

##### 3.1.3.3 10 Mb/s Serial Mode Operation

For applications that have external ENDECs for 10 Mb/s operation, the DP83840 accepts Manchester encoded serial data on the TXD[0] input and provides Manchester encoded serial data output on RXD[0] with a clock rate of 10 MHz.

This mode is selected by setting the 10BT\_SER bit in the 10BTSR (bit 9, register address 1Bh). The default value for this bit is set by the 10BTSER pin 98 at power-up/reset.

### 3.2 MII SERIAL MANAGEMENT REGISTER ACCESS

The MII specification defines a set of thirty-two 16-bit status and control registers that are accessible through the serial management data interface pins MDC and MDIO. The DP83840 implements all the required MII registers and a subset of optional registers. The registers are fully described in Section 4. The serial management access protocol is described below.

#### 3.2.1 Serial Management Access Protocol

The serial control interface consists of two pins, Management Data Clock (MDC) and Management Data Input/Output (MDIO). MDC has a maximum clock rate of 2.5 MHz. The MDIO line is bi-directional and may be shared by up to 32 devices. The MDIO frame format is shown in Table 1.



### 3.0 Functional Description (Continued)

The MDIO pin requires a pull-up resistor (4.7 kΩ) which, during IDLE condition, will pull MDIO high. Prior to initiating any transaction, the station management entity sends a sequence of 32 contiguous logic ones on MDIO to provide the DP83840 with a sequence that can be used to establish synchronization.

The DP83840 waits until it has received this sequence before responding to any other transaction.

The Start code is indicated by a <01> pattern. This assures the MDIO line transitions from the default idle line state.

Turnaround is an idle bit time inserted between the Register Address field and the Data field. To avoid contention, no device actively drives the MDIO signal during the first bit of Turnaround during a read transaction. The addressed DP83840 drives the MDIO with a zero for the second bit of Turnaround and follows this with the required data. Figure 2 shows the timing relationship between MDC and the MDIO as driven/received by the Station Management Entity (STA) and the DP83840 (PHY) for a typical register read access.

For write transactions, the station management entity writes data to an addressed DP83840 eliminating the requirement for MDIO Turnaround. The Turnaround time is filled by the management entity inserting <10> for these two bits. Figure 3 shows the timing relationship for a typical MII register write access.

#### 3.2.2 PHY Address Sensing

The DP83840 can be set to respond to any of the possible 32 PHY addresses. Note that each DP83840 connected to a common MII must have a unique address.

The DP83840 provides five PHY address pins, the state of which are latched into the PHY Address Register (PAR) at system power-up/reset. These pins are described in Section 2.8.

#### 3.2.3 MII Management

The MII may be used to connect PHY devices to MAC or repeater devices in 10/100 Mb/s systems.

The management interface of the MII allows the configuration and control of multiple PHY devices, the gathering of status and error information, and the determination of the type and abilities of the attached PHY(s).

#### 3.3 100BASE-X TRANSMITTER

The 100BASE-X transmitter consists of functional blocks which convert synchronous 4-bit nibble data, as provided by the MII, to a scrambled 125 Mb/s serial data stream. This data stream may be routed either to a twisted pair PMD such as the DP83223 TWISTER for 100BASE-TX signaling, or to an optical PMD for 100BASE-FX applications. The block diagram in Figure 4 provides an overview of each functional block within the 100BASE-X transmit section.

The Transmitter section consists of the following functional blocks:

- Symbol Encoder and Injection block (bypass option)
- Scrambler block (bypass option)
- NRZ to NRZI encoder block (bypass option)

The bypass option for each of the functional blocks within the 100BASE-X transmitter provides flexibility for applications such as 100 Mb/s repeaters where data conversion is not always required.

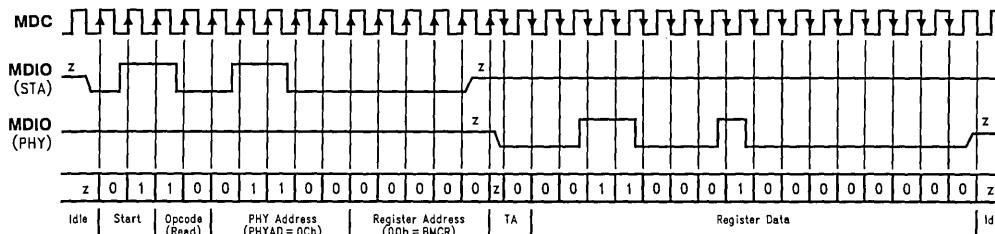


FIGURE 2. Typical MDC/MDIO Read Operation

TL/F/12388-34

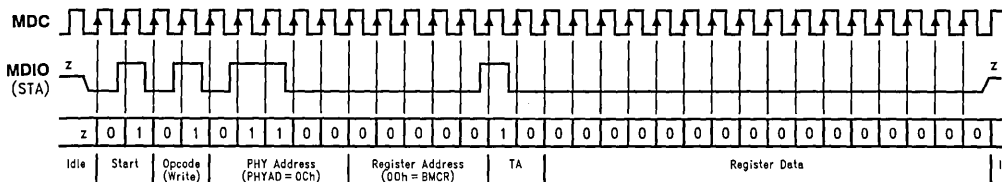


FIGURE 3. Typical MDC/MDIO Write Operation

TL/F/12388-35

TABLE I. MII Management Serial Protocol

|                                |                                                                              |
|--------------------------------|------------------------------------------------------------------------------|
| MII Management Serial Protocol | <idle> <start> <op code> <device addr> <reg addr> <turnaround> <data> <idle> |
| Read Operation                 | <idle> <01> <10> <AAAAA> <RRRRR> <Z0> <xxxx xxxx xxxx xxx> <idle>            |
| Write Operation                | <idle> <01> <01> <AAAAA> <RRRRR> <10> <xxxx xxxx xxxx xxx> <idle>            |

### 3.0 Functional Description (Continued)

#### 3.3.1 100 Mb/s Transmit State Machine

The DP83840 implements the 100BASE-X transmit state machine diagram as given in the IEEE 802.3u Standard, Clause 24.

#### 3.3.2 Symbol Encoding and Injection

The symbol encoder converts 4 bit (4B) nibble data generated by the MAC into 5 bit (5B) symbols for transmission. This conversion is required to allow control symbols to be combined with data symbols. Refer to Table II for 4B to 5B symbol mapping details.

The symbol encoder substitutes the first 8 bits of the MAC preamble with a J/K symbol pair (11000 10001). The symbol encoder continues to replace subsequent 4B codes with corresponding 5B symbols. At the end of the transmit packet, the symbol encoder injects the T/R symbol pair indicating end of frame.

The symbol encoder continuously injects IDLE symbols into the transmit data stream until the next transmit packet is detected.

#### 3.3.3 Scrambler

The scrambler is required to control the radiated emissions at the media connector and on the twisted pair cable (for 100BASE-TX applications). By scrambling the data, the total energy launched onto the cable is randomly distributed over a wide frequency range. Without the scrambler, energy levels at the PMD and on the cable would peak at frequencies related to repeating 5B sequences (i.e., continuous transmission of IDLEs).

The scrambler is configured as a closed loop linear feedback shift register (LFSR) with an 11-bit polynomial. The output of the closed loop LFSR is combined with the NRZ 5B data from the symbol encoder via an X-OR logic function. The result is a scrambled data stream with sufficient randomization to decrease radiated emissions at certain frequencies by as much as 20 dB.

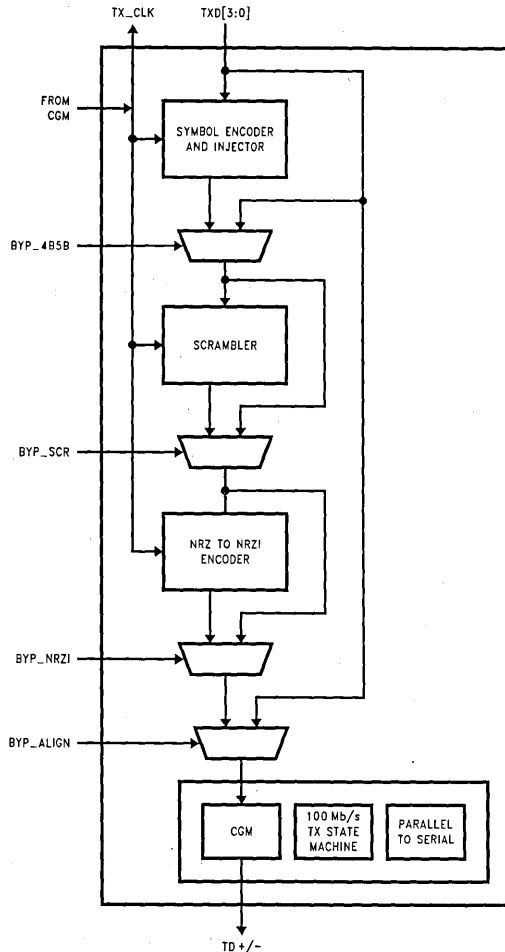


FIGURE 4. 100BASE-X Transmitter

TL/F/12388-4

## 3.0 Functional Description (Continued)

### 3.3.4 NRZ to NRZI Encoder

After the transmit data stream is scrambled, the data must be NRZI encoded in order to comply with the TP-PMD standard for 100BASE-TX transmission over Category-5 unshielded twisted pair cable.

### 3.4 100BASE-X RECEIVER

The 100BASE-X receiver consists of several functional blocks which are required to recover and condition the 125 Mb/s receive data stream as specified by the IEEE 802.3u Standard. The 125 Mb/s receive data stream may originate from a twisted pair transceiver such as the DP83223 TWISTER in a 100BASE-TX application. Alternatively, the receive data stream may be generated by an optical receiver as in a 100BASE-FX application. The block diagram in *Figure 5* provides an overview of each functional block within the 100BASE-X receive section.

The Receiver block consists of the following functional blocks:

- Clock Recovery block
- NRZI to NRZ decoder block (bypass option)
- Descrambler block (bypass option)
- Symbol Alignment block (bypass option)
- 5B/4B Symbol Decoder block (bypass option)
- Collision Detect block
- Carrier Sense block
- Stream Decoder block
- 100 Mb/s Receive State Machine

The bypass option for each of the functional blocks within the 100BASE-X receiver provides flexibility for applications such as 100 Mb/s repeaters where data conversion is not always required.

#### 3.4.1 Clock Recovery

The Clock Recovery Module (CRM) accepts 125 Mb/s scrambled NRZI data stream from an external PMD receiver (DP83223). The CRM locks onto the 125 Mb/s data stream and extracts a 125 MHz reference clock. The extracted and synchronized clock and data are used as required by the synchronous receive operations.

The CRM is implemented using an advanced digital Phase Locked Loop (PLL) architecture that replaces sensitive analog circuits. Using digital PLL circuitry allows the DP83840 to be manufactured and specified to tighter tolerances.

### 3.4.2 NRZI to NRZ

In a typical application the NRZI to NRZ decoder is required in order to present NRZ formatted data to the descrambler.

The receive data stream, as recovered by the PMD receiver, is in NRZI format, therefore the data must be decoded to NRZ before reaching the descrambler. With the receive data in NRZ format, the descrambler can properly synchronize to the scrambled data.

### 3.4.3 Descrambler

A 5-bit parallel (symbol wide) descrambler is used to descramble the receive NRZ data. To reverse the data scrambling process, the descrambler has to generate an identical data scrambling sequence (N) in order to recover the original unscrambled data (UD) from the scrambled data (SD) as represented in the equations:

$$SD = UD \oplus N \quad UD = SD \oplus N$$

Synchronization of the descrambler to the original scrambling sequence (N) is achieved based on the knowledge that the incoming scrambled data stream consists of scrambled IDLE data. After the descrambler has recognized sufficient IDLE symbols, where an IDLE symbol in 5B NRZ is equal to five consecutive ones (11111), it will synchronize to the receive data stream and generate unscrambled data in the form of unaligned 5B symbols.

In order to maintain synchronization, the descrambler must continuously monitor the validity of the unscrambled data that it generates. To ensure this, a line state monitor and a hold timer are used to constantly monitor the synchronization status. Upon synchronization of the descrambler the hold timer starts a 722  $\mu$ s countdown. Upon detection of sufficient IDLE symbols within the 722  $\mu$ s period, the hold timer will reset and begin a new countdown. This monitoring operation will continue indefinitely given a properly operating network connection with good signal integrity. If the line state monitor does not recognize sufficient unscrambled IDLE symbols within the 722  $\mu$ s period, the entire descrambler will be forced out of the current state of synchronization and reset in order to re-acquire synchronization.

### 3.4.4 Symbol Alignment

The Symbol Alignment module operates on unaligned 5-bit data from the descrambler (if the descrambler is not bypassed) and converts it into 5B symbol data (5 bits). Symbol alignment occurs after the J/K symbol pair is detected. Once the J/K symbol pair (11000 10001) is detected, subsequent data is aligned on a fixed boundary.

### 3.0 Functional Description (Continued)

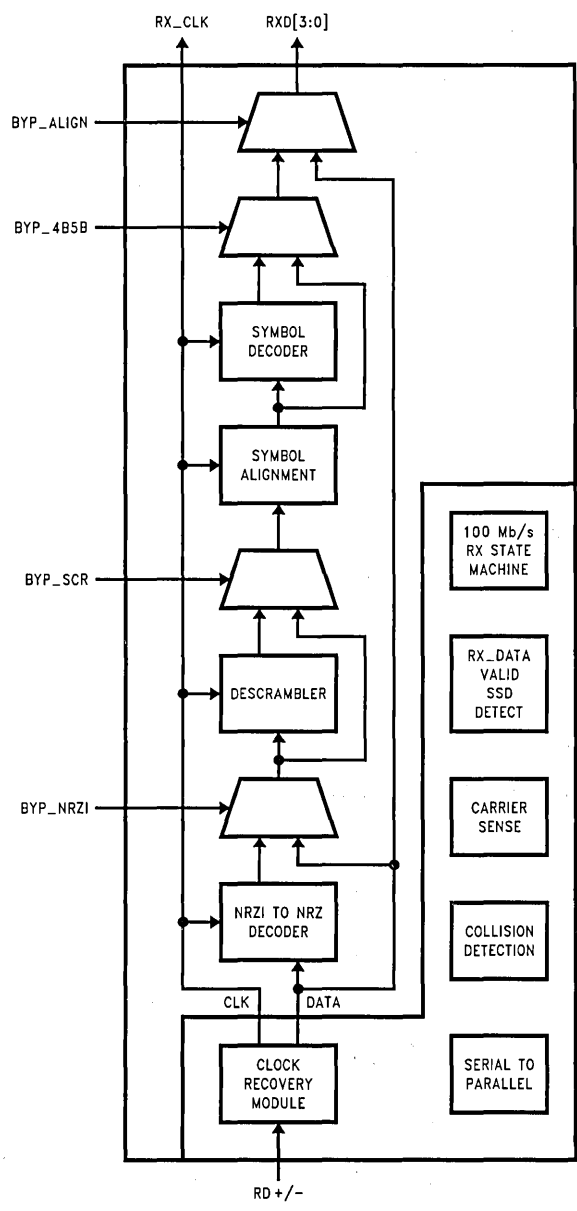
**TABLE II. 4B5B Symbol Encoding/Decoding**

| Symbol                        | 5B Symbol Code | 4B Nibble Code                         |
|-------------------------------|----------------|----------------------------------------|
| <b>DATA CODES</b>             |                |                                        |
| 0                             | 11110          | 0000                                   |
| 1                             | 01001          | 0001                                   |
| 2                             | 10100          | 0010                                   |
| 3                             | 10101          | 0011                                   |
| 4                             | 01010          | 0100                                   |
| 5                             | 01011          | 0101                                   |
| 6                             | 01110          | 0110                                   |
| 7                             | 01111          | 0111                                   |
| 8                             | 10010          | 1000                                   |
| 9                             | 10011          | 1001                                   |
| A                             | 10110          | 1010                                   |
| B                             | 10111          | 1011                                   |
| C                             | 11010          | 1100                                   |
| D                             | 11011          | 1101                                   |
| E                             | 11100          | 1110                                   |
| F                             | 11101          | 1111                                   |
| <b>IDLE AND CONTROL CODES</b> |                |                                        |
| H                             | 00100          | Halt Symbol—To be transmitted on TX_ER |
| I                             | 11111          | Inter Packet Idle Symbol—0000*         |
| J                             | 11000          | First Start of Packet Symbol—0101*     |
| K                             | 10001          | Second Start of Packet Symbol—0101*    |
| T                             | 01101          | First End of Packet Symbol—0000*       |
| R                             | 00111          | Second End of Packet Symbol—0000*      |
| <b>INVALID CODES</b>          |                |                                        |
| V                             | 00000          | 0110 or 0101**                         |
| V                             | 00001          | 0110 or 0101**                         |
| V                             | 00010          | 0110 or 0101**                         |
| V                             | 00011          | 0110 or 0101**                         |
| V                             | 00101          | 0110 or 0101**                         |
| V                             | 00110          | 0110 or 0101**                         |
| V                             | 01000          | 0110 or 0101**                         |
| V                             | 01100          | 0110 or 0101**                         |
| V                             | 10000          | 0110 or 0101**                         |
| V                             | 11001          | 0110 or 0101**                         |

\*Control symbols I, J, K, T and R in data fields will be mapped as invalid codes, together with RX\_ER asserted.

\*\*Normally, invalid codes (V) are mapped to 6h on RXD[3:0] with RX\_ER asserted. If the CODE\_ERR bit in the LBREMR (bit 4, register address 18h) is set, the invalid codes are mapped to 5h on RXD[3:0] with RX\_ER asserted.

### 3.0 Functional Description (Continued)



TL/F/12388-5

FIGURE 5. 100BASE-X Receiver

### 3.0 Functional Description (Continued)

#### 3.4.5 Symbol Decoder

The Symbol Decoder functions as a look up table that translates incoming 5B symbols into 4B nibbles. The Symbol Decoder first detects the J/K symbol pair preceded by IDLE symbols and replaces the symbol with MAC preamble. Specifically, the J/K 10-bit symbol pair is replaced by the nibble pair 1010 1010. All subsequent 5B symbols are converted to the corresponding 4B nibbles for the duration of the entire packet. This conversion ceases upon the detection of the T/R symbol pair denoting the End of Stream Delimiter (ESD).

#### 3.4.6 Collision Detect (Half Duplex)

Half Duplex collision detection for 100 Mb/s follows the model of 10BASE-T. Collision detection is indicated by the COL pin of the MII.

For Full Duplex applications the COL signal is never asserted.

#### 3.4.7 Carrier Sense

Carrier Sense (CRS) is asserted upon the detection of two non-contiguous zeros occurring within any 10-bit boundary of the receive data stream.

The carrier sense function is independent of symbol alignment. For 100 Mb/s Half Duplex operation, CRS is asserted during either packet transmission or reception.

For 100 Mb/s Full Duplex operation, CRS is asserted only during packet reception.

When the IDLE symbol pair is detected in the receive data stream, CRS is deasserted.

In REPEATER mode (pin 47/bit 12, register address 17h), CRS is only asserted due to receive activity.

#### 3.4.8 100 Mb/s Receive State Machine

The DP83840 implements the 100BASE-X receive state machine diagram as given in ANSI/IEEE Standard 802.3u, Clause 24.

#### 3.5 CLOCK GENERATION MODULE

The Clock Generation Module (CGM) within the DP83840 can be configured for several different applications. This offers the flexibility of selecting a clocking scheme that is best suited for a given design.

This section describes the operation of the CGM from a device perspective as well as its applications within a system such as an adapter or repeater.

##### 3.5.1 Device Requirements

For 100 Mb/s operation the DP83840 requires either a 50 MHz reference at the OSCIN pin or a 25 MHz reference at the REFIN pin.

If 10BASE-T operation and/or Auto-Negotiation functions are required, a 20 MHz reference is also required. This can be derived either internally from the 50 MHz reference or externally from a 20 MHz crystal or oscillator.

The DP83840 will accept various clock reference inputs. Each of these is described as follows.

##### 3.5.1.1 Single 50 MHz Reference

A 50 MHz oscillator can be used to drive the OSCIN input. This reference is internally divided by two and then routed to the CLK25M output pin. By connecting the CLK25M output directly to the REFIN input pin, the 25 MHz reference is allowed to drive the 100 Mb/s module. The 50 MHz signal is also divided by 2.5 internally to provide the 20 MHz reference directly to the 10 Mb/s module. This option is shown in Figure 6. It should be noted that the  $\div 2.5$  circuitry adds some jitter to the 10 Mb/s performance.

The 10BASE-T module within the DP83840 will automatically switch to the 20 MHz reference (sourced by the internal  $\div 2.5$  circuit) upon detection of inactivity on the X1 input pin. When not in use, the X1 input pin should be tied to  $V_{CC}$  either directly or via a 4.7 k $\Omega$  pull-up resistor.

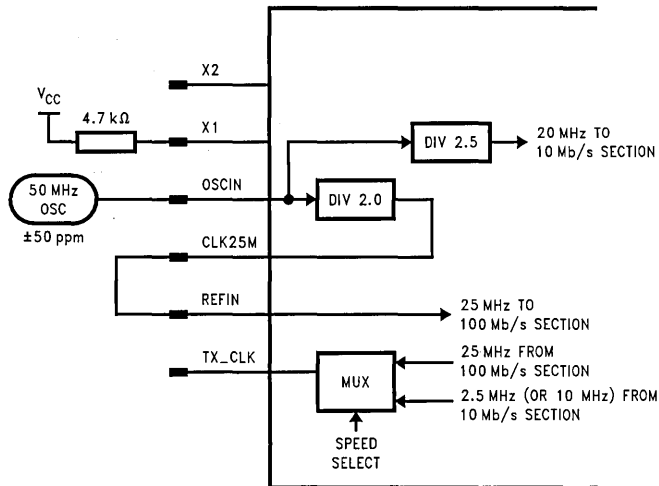


FIGURE 6. Single 50 MHz Reference

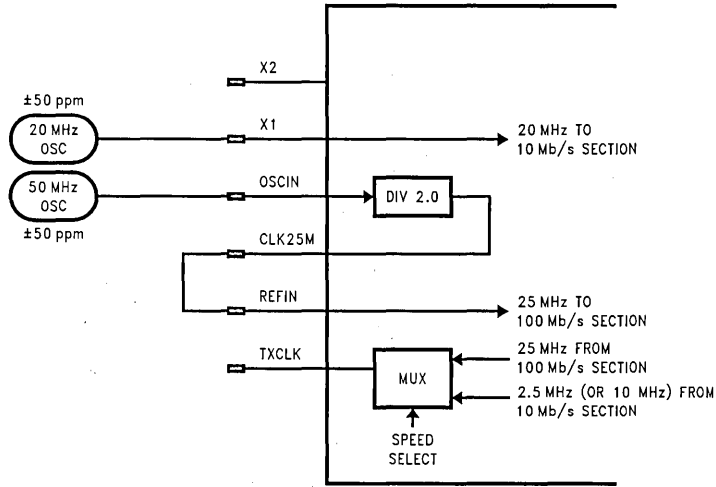
TL/F/12388-6

### 3.0 Functional Description (Continued)

#### 3.5.1.2 50 MHz and 20 MHz Reference

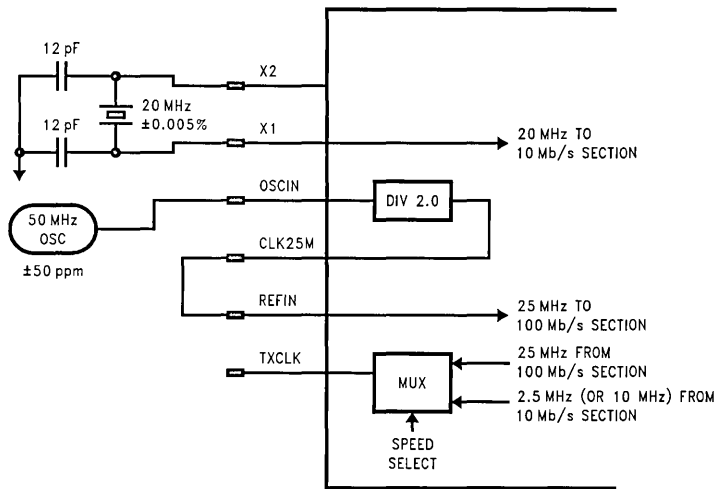
For improved jitter performance in the 10 Mb/s module, an external 20 MHz oscillator can be used to drive the X1 pin. Alternatively, a 20 MHz crystal network can be connected across pins X1 and X2 to provide the required reference for

the 10 Mb/s module. The 100 Mb/s module must still receive a 25 MHz reference which can be provided by a 50 MHz oscillator as described in 3.5.1.1. This option is shown in *Figure 7* (20 MHz oscillator module) and *Figure 8* (20 MHz crystal) below.



TL/F/12388-7

FIGURE 7. 50 MHz and 20 MHz Reference



TL/F/12388-8

FIGURE 8. 50 MHz Reference and 20 MHz Crystal

### 3.0 Functional Description (Continued)

#### 3.5.1.3 25 MHz and 20 MHz Reference

A 25 MHz reference, either from an oscillator or a system clock can directly drive the 100 Mb/s module via the REFIN input.

A separate 20 MHz reference from either an oscillator or a crystal network must be provided to the X1 and X2 inputs as described in 3.5.1.2. This option is shown in *Figure 9* below.

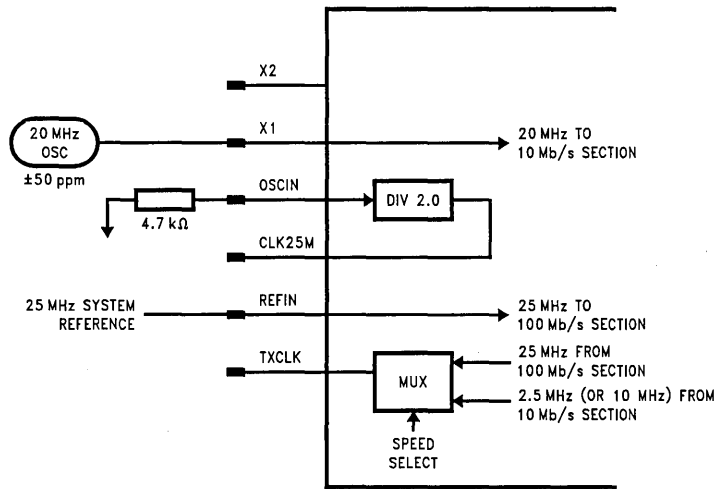


FIGURE 9. 25 MHz and 20 MHz Reference

TL/F/12388-9



### 3.0 Functional Description (Continued)

#### 3.5.2 System Clocking

The DP83840 clock options help to simplify single port adapter designs as well as multi-port repeaters. The TX\_CLK allows MII data to be received in either parallel or serial modes as described in Section 3.1.3. The standard MII interface clock rate options are as follows:

TX\_CLK = 25 MHz for 100 Mb/s nibble mode

TX\_CLK = 2.5 MHz for 10 Mb/s nibble mode

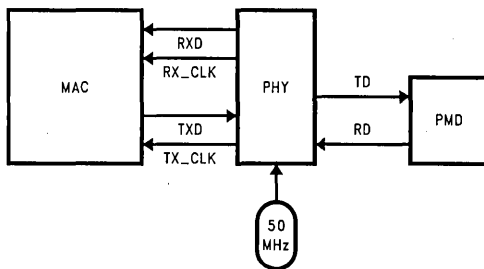
Additionally, the DP83840 provides:

TX\_CLK = 10 MHz for 10 Mb/s serial mode

##### 3.5.2.1 Adapter Clock Distribution Example

In most single port adapter applications, where only one DP83840 is required, providing a single 50 MHz oscillator reference is sufficient for deriving the required MAC and PHY layer clocks. Based on the 50 MHz reference, the DP83840 can generate its own internal 20 MHz reference for the 10 Mb/s module. Additionally, the DP83840 can generate the required 25 MHz reference for its 100 Mb/s module.

During 100 Mb/s operation the 25 MHz reference generated by the DP83840 is available at the TX\_CLK output pin. This can be used for synchronization with the MAC layer device. During 10 Mb/s operation the TX\_CLK pin sources either a 2.5 MHz or 10 MHz reference to the MAC layer device. *Figure 10* provides an example of the clock distribution in a typical node design based on the DP83840.

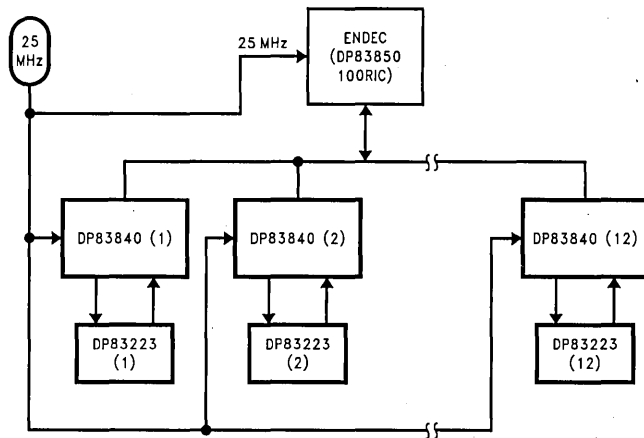


TL/F/12388-10

**FIGURE 10. Typical Adapter Clock and Data Interconnections**

##### 3.5.2.2 Repeater Clock Distribution Example

The clock distribution within a multi-port repeater can be designed in a variety of ways. *Figure 11* provides a simplified example of a timing distribution scheme in a 100 Mb/s only repeater design. It should be noted that in order to support Auto-Negotiation, a 20 MHz reference would be required for each DP83840 device. Due to the demanding timing constraints required to maintain standards compliance, great care must be taken in the design and layout of a multi-port repeater system. The example provided in *Figure 11* illustrates interconnection only and should not be considered as a final working design.



TL/F/12388-11

**FIGURE 11. Typical 100 Mb/s Repeater Clock Interconnection**

## 3.0 Functional Description (Continued)

### 3.6 CLOCK RECOVERY MODULE

The Clock Recovery Module (CRM) is part of the 100 Mb/s receive channel. The 10 Mb/s clock recovery is independent from the CRM.

The CRM contains a Phase Locked Loop that tracks the signal frequency of the incoming 125 Mb/s data stream at the RD+/- inputs. The CRM extracts a synchronous 125 MHz clock from this data (the data rate on the cable is 125 Mb/s due to 4B5B encoding). The CRM obtains its initial frequency and stability from its own internal VCO and then adjusts the frequency as required to match the incoming data stream frequency. The CRM maintains control of the PLL's loop gain to minimize the lock time as well as to minimize the jitter after phase lock has been acquired.

When the Signal Detect (SD+/-) inputs become active, the CRM attempts to acquire lock. The CRM loses lock when the SD+/- inputs become de-asserted.

The CRM generates a 125 MHz clock synchronous with the receive data stream and presents both the clock and data to the rest of the 100 Mb/s receive section. The CRM is not synchronous with the local clock present at the REFIN input to the CGM.

The RX\_CLK signal at the MII interface is derived from the CRM 125 Mb/s clock during 100 Mb/s operation. The RX\_CLK frequency is set to 25 MHz for nibble-wide receive data passing to the MAC and/or ENDEC.

### 3.7 10BASE-T TRANSCEIVER MODULE

The 10BASE-T Transceiver Module is IEEE 802.3 compliant. It includes the receiver, transmitter, collision, heartbeat, loopback, jabber, and link integrity functions, as defined in the standard. An external filter is not required on the 10BASE-T interface since this is integrated inside the DP83840.

#### 3.7.1 Operational Modes

The DP83840 has 2 basic 10 Mb/s operational modes:

- Half Duplex mode
- Full Duplex mode

##### 3.7.1.1 Half Duplex Mode

In Half Duplex mode the DP83840 functions as a standard IEEE 802.3 10BASE-T transceiver with fully integrated filtering.

##### 3.7.1.2 Full Duplex Mode

In Full Duplex mode the DP83840 is capable of simultaneously transmitting and receiving without asserting the collision signal. The DP83840's 10 Mb/s ENDEC is designed to encode and decode simultaneously.

#### 3.7.2 Smart Squelch

The Smart Squelch is responsible for determining when valid data is present on the differential receive inputs (RXI±). The DP83840 implements an intelligent receive squelch on the RXI± differential inputs to ensure that impulse noise on the receive inputs will not be mistaken for a valid signal. Smart squelch operation is independent of the 10BASE-T operational mode.

The squelch circuitry employs a combination of amplitude and timing measurements (as specified in the IEEE 802.3 10BASE-T standard) to determine the validity of data on the twisted pair inputs.

The receive squelch threshold level can be lowered for use in longer cable or STP applications. This is achieved by setting the LSS bit in the 10BTCCR (bit 2, register address 1Ch).

#### 3.7.3 Collision Detection

For Half Duplex, a 10BASE-T collision is detected when the receive and transmit channels are active simultaneously. Collisions are reported by the COL signal on the MII.

If the ENDEC is transmitting when a collision is detected, the collision is not reported until seven bits have been received while in the collision state. This prevents a collision being reported incorrectly due to noise on the network. The COL signal remains set for the duration of the collision.

When heartbeat is enabled, approximately 1 μs after the transmission of each packet a Signal Quality Error (SQE) signal of approximately 10 bit times is generated (internally) to indicate successful transmission. SQE is reported as a pulse on the COL signal of the MII.

#### 3.7.4 Link Pulse Detection/Generation

The link pulse generator produces pulses as defined in the IEEE 802.3 10BASE-T standard. Each link pulse is nominally 100 ns in duration and is transmitted every 16 ms ± 8 ms, in the absence of transmit data.

Link pulse is used to check the integrity of the connection with the remote end. If valid link pulses are not received, the link detector disables the 10BASE-T twisted pair transmitter, receiver and collision detection functions.

When the link integrity function is disabled, the 10BASE-T transceiver will operate regardless of the presence of link pulses.

In 10 Mb/s ENDEC loopback mode (bit 11, register address 18h), transmission and reception paths can be tested regardless of the incoming link status.

#### 3.7.5 Jabber Function

The Jabber function monitors the DP83840's output and disables the transmitter if it attempts to transmit a longer than legal sized packet. A jabber timer monitors the transmitter and disables the transmission if the transmitter is active for greater than approximately 26 ms.

Once disabled by the Jabber function, the transmitter stays disabled for the entire time that the ENDEC module's internal transmit enable is asserted. This signal has to be de-asserted for approximately 750 ms (the "unjab" time) before the Jabber function re-enables the transmit outputs.

#### 3.7.6 Transmit Outputs

There are two pairs of 10BASE-T output signals. One pair for UTP cable (TXU+/-) and one pair for STP cable (TXS+/-).

Selection between 100Ω UTP and 150Ω STP cable operation is accomplished using the UTP/STP bit in the 10BASE-T Configuration Register (bit 3, register address 1Ch). Only one set of outputs is active at a time. Selecting UTP will TRI-STATE STP and vice versa.

The TXU+/- and TXS+/- outputs of the DP83840 are internally filtered and require no additional external filtering. See Section 3.7.11 for more details.

#### 3.7.7 Status Information

10BASE-T Status Information is available on the LED output pins of the DP83840. Transmit activity, receive activity, link

### 3.0 Functional Description (Continued)

status, link polarity and collision activity information is output to the five LED output pins (LED1 to LED5). See Section 2.6 for more information on these outputs.

If required the LED outputs can be used to provide digital status information to external circuitry.

The Link LED output (LED3, pin #38) indicates Good Link status for both 10 and 100 Mb/s modes. In Half Duplex 10BASE-T mode, LED3 indicates link status.

The link integrity function can be disabled. When disabled, the transceiver will operate regardless of the presence of link pulses and the Link LED will stay lit continuously.

#### 3.7.8 Automatic Link Polarity Detection

The DP83840's 10BASE-T Transceiver Module incorporates an automatic link polarity detection circuit. When seven consecutive link pulses or three consecutive receive packets with inverted End-of-Packet pulses are received, bad polarity is reported.

A polarity reversal can be caused by a wiring error at either end of the UTP/STP cable, usually at the Main Distribution Frame (MDF) or patch panel in the wiring closet.

The bad polarity condition is latched and the LED4 output is asserted. The DP83840's 10BASE-T Transceiver Module corrects for this error internally and will continue to decode received data correctly. This eliminates the need to correct the wiring error immediately.

#### 3.7.9 10BASE-T Internal Loopback

When the 10BT\_LPBK bit in the LBREMR (bit 11, register address 18h) is set, 10BASE-T transmit data is looped back in the ENDEC to the receive channel. The transmit drivers and receive input circuitry are disabled in transceiver loopback mode, isolating the transceiver from the network.

Loopback is used for diagnostic testing of the data path through the transceiver without transmitting on the network or being interrupted by receive traffic.

#### 3.7.10 Transmit and Receive Filtering

External 10BASE-T filters are not required when using the DP83840 as the required signal conditioning is integrated.

Only isolation/step-up transformers and impedance matching resistors are required for the 10BASE-T transmit and receive interface. The internal transmit filtering ensures that all the harmonics in the transmit signal are attenuated by at least 30 dB.

#### 3.7.11 Typical Node Application

An example of the 10BASE-T interface is shown in *Figure 12*. The TXS+/- signals are used for STP and the TXU+/- signals for UTP. Standard UTP applications do not require connection of the TXS+/- outputs. The output resistor values are chosen to match the transmit output impedance to the impedance of the twisted pair cable.

The DP83840 10BASE-T outputs require a 1:2 step-up isolation transformer in order to match the cable impedance. The 10BASE-T inputs require a 1:1 isolation transformer and appropriate line termination. Refer to *Figure 12*.

### 3.8 IEEE 1149.1 CONTROLLER

The IEEE 1149.1 standard defines a test access port and boundary-scan architecture for digital integrated circuits and for the digital portions of mixed analog/digital integrated circuits.

The standard provide a solution for testing assembled printed circuit boards and other products based on highly complex digital integrated circuits and high-density surface-mounting assembly techniques. It also provides a means of accessing and controlling design-for-test features built into the digital integrated circuits. Such features include internal scan paths and self-test functions as well as other features intended to support service applications in the assembled product. The IEEE 1149.1 Boundary Scan Architecture document should be referenced for additional detail.

The circuitry defined by this standard allows test instructions and associated data to be input serially into a device. The instruction execution results are output serially.

The DP83840 reserves five pins, called the Test Access Port (TAP), to provide test access: TMS, TCK, Test Data Input (TDI), Test Data Output (TDO) and Test Reset (TRST). These signals are described in Section 2.7. To ensure race-free operation all input and output data is synchronous to the test clock (TCK). TAP input signals (TMS and TDI) are clocked into the test logic on the rising edge of TCK while output signal (TDO) is clocked on the falling edge.

#### 3.8.1 Test Logic

The IEEE 1149.1 Test Logic consists of a Test Access Port (TAP) controller, an instruction register, and a group of test data registers including Bypass, Device Identification and Boundary Scan registers.

The TAP controller is a synchronous 16 state machine that responds to changes at the TMS and TCK signals.

This controls the sequence of operations by generating clock and control signals to the instruction and test data registers. The control signals switch TDI and TDO between instruction and test data registers.

The DP83840 implements 4 basic instructions: ID\_Code, bypass, Sample/Preload and Extest. Upon reset, the ID\_Code instruction is selected by default. If the ID\_Code instruction is not supported, the bypass instruction is selected instead.

##### 3.8.1.1 ID\_Code Instruction

The ID\_Code instruction allows users to select the 32-bit IDCODE register and interrogate the contents which consist of the manufacturer's ID, part ID and the version number.

##### 3.8.1.2 Bypass Instruction

The bypass instruction uses the bypass register. The bypass register contains a single shift-register stage and is used to provide a minimum length serial path between the TDI and TDO pins of the DP83840 when test operation is not required. This allows more rapid movement of test data to and from other testable devices in the system.

##### 3.8.1.3 Sample/Preload Instruction

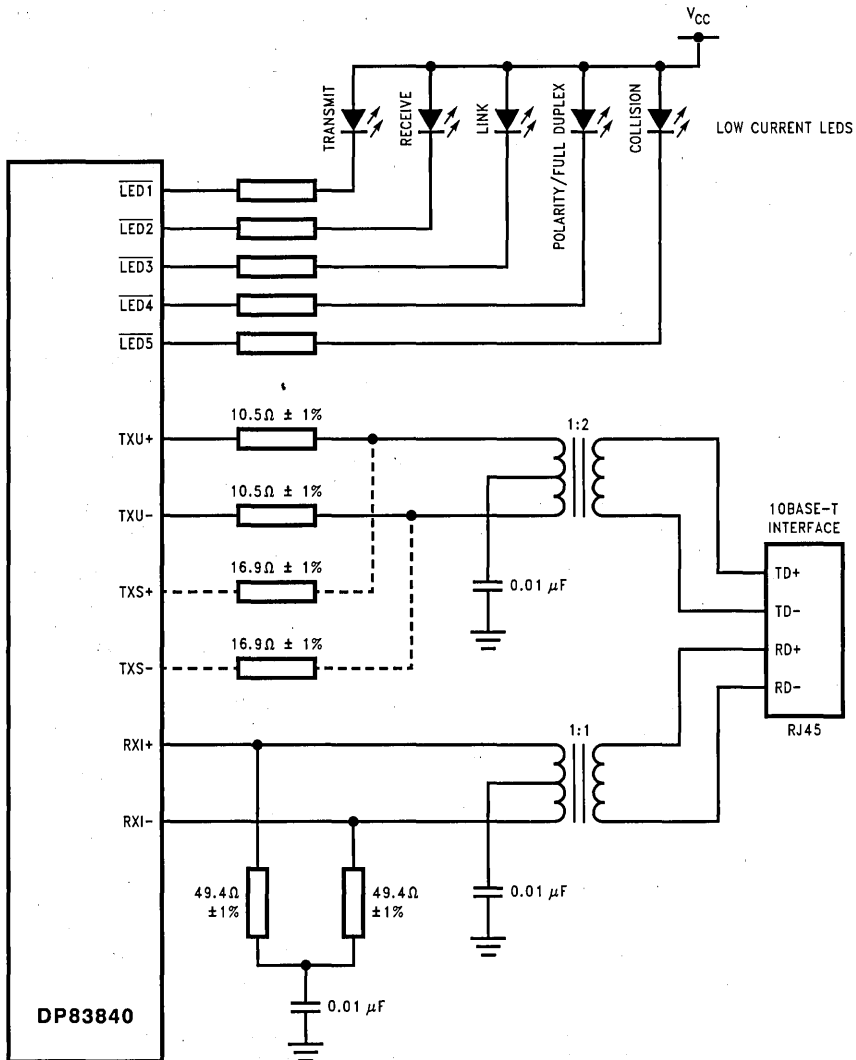
The Sample/Preload instruction allows scanning of the boundary-scan register without causing interference to the normal operation of the on-chip system logic.

Two functions are performed when this instruction is selected.

### 3.0 Functional Description (Continued)

Sample allows a snapshot to be taken of the data flowing from the system pins to the on-chip test logic or vice versa, without interfering with normal operation. The snapshot is taken on the rising edge of TCK in the Capture-DR controller state, and the data can be viewed by shifting through the component's TDO output.

While sampling and shifting data out through TDO for observation, preload allows an initial data pattern to be shifted in through TDI and to be placed at the latched parallel output of the boundary-scan register cells which are connected to system output pins. This ensures that known data is driven through the system output pins upon entering the Extest instruction.



Note: Resistors from TXS+/- outputs can be added if STP cable support is required.

TL/F/12388-12

FIGURE 12. Typical 10BASE-T (UTP) Node Application

### 3.0 Functional Description (Continued)

Without Preload, indeterminate data would be driven until the first scan sequence has been completed. The shifting of data for the Sample and Preload phases can occur simultaneously. While data capture is being shifted out, the preload data can be shifted in.

#### 3.8.1.4 Extest Instruction

The Extest instruction allows circuitry external to the DP83840 (typically the board interconnections) to be tested. Prior to executing the Extest instruction, the first test stimulus to be applied will be shifted into the boundary-scan registers using the Sample/Preload instruction. Thus, when the change to the Extest instruction takes place, known data will be driven immediately from the DP83840 to its external connections.

This provides stimulus to the system input pins of adjacent devices on the assembled printed circuit boards. *Figure 13* below illustrates the IEEE 1149.1 architecture.

#### 3.8.2 Device Testing

IEEE 1149.1 provides a simple solution for testing many of the standard static pin parametrics. Reasonably accurate limits may be tested as a functional pattern.

The IEEE 1149.1 test circuitry is tested itself as a consequence of testing pin parametrics. Specific tests are:

TRI-STATE conditions of TDO when serial shift between TDI and TDO is not selected

Input leakage of TCK, TMS, TDI and TRST

Output has TRI-STATE leakage of TDO

Opens and shorts of TCK, TMS, TDI, TRST, and TDO

IDCODE register, the bypass register and the TAP controller state machine sequences

Open and shorted pins can be identified by placing an alternating bit pattern on the I/O pins. Any shorted bond wires would either cause an input to be misinterpreted in the inputs scan phase, or the test comparator would fail an output during data scan.

Repeating the test with the inverse bit pattern provides coverage of V<sub>CC</sub> and GND short/open circuits.

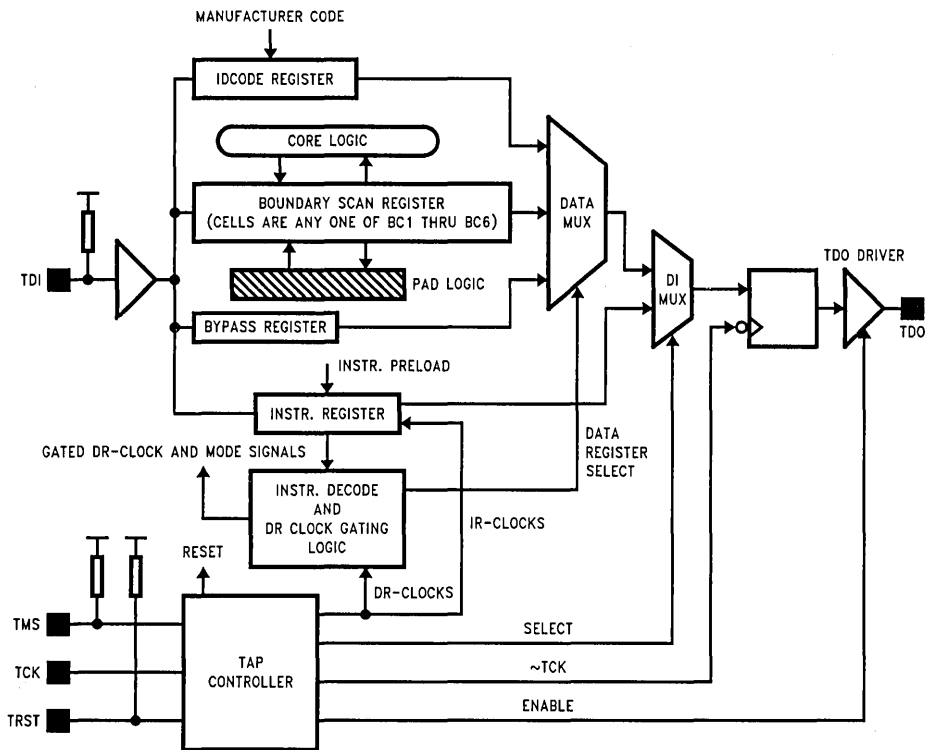


FIGURE 13. IEEE 1149.1 Architecture

TL/F/12388-13

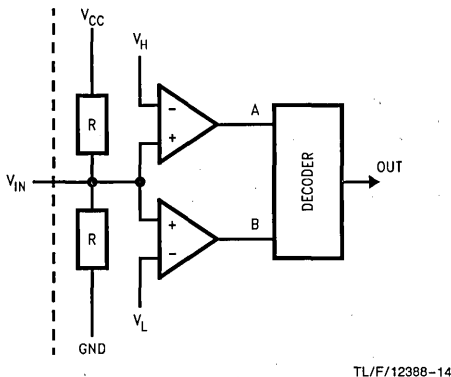
### 3.0 Functional Description (Continued)

#### 3.9 IEEE 802.3u AUTO-NEGOTIATION

The Auto-Negotiation function provides a mechanism for exchanging configuration information between two ends of a link segment and automatically selecting the highest performance mode of operation supported by both devices. Fast Link Pulses (FLP) Bursts provide the signaling used to communicate Auto-Negotiation abilities between two devices at each end of a link segment. For further detail regarding Auto-Negotiation, refer to clause 28 of the IEEE 802.3u specification. The DP83840 supports four different Ethernet protocols, so the inclusion of Auto-Negotiation ensures that the highest performance protocol will be selected based on the ability of the Link Partner. The Auto-Negotiation function within the DP83840 can be controlled either by internal register access or by use of the AN1 and AN0 (pins 46 and 95).

#### 3.9.1 Auto-Negotiation Pin Control

The state of AN0 and AN1 determines whether the DP83840 is forced into a specific mode or Auto-Negotiation will advertise a specific ability or set of abilities as given in Table III. Pins AN0 and AN1 are implemented as tri-level control pins which are configured by connecting them to V<sub>CC</sub>, GND or by leaving them unconnected (refer to Figure 14). The state of AN0 and AN1 determines the Auto-Negotiation mode upon power-up/reset. This state is not reflected in the BCMR. The Auto-Negotiation function selected at power-up/reset can be changed at any time by writing to the Basic Mode Command Register (BCMR) at address 00h.



| V <sub>IN</sub>     | A | B | OUT |
|---------------------|---|---|-----|
| 0V                  | L | L | L   |
| V <sub>CC</sub> ÷ 2 | L | H | M   |
| V <sub>CC</sub>     | H | H | H   |

FIGURE 14. Tri-Level Pin Control

#### 3.9.2 Auto-Negotiation Register Control

When Auto-Negotiation is enabled, the DP83840 transmits the abilities programmed into the Auto-Negotiation Advertisement Register (ANAR) at address 04h via FLP Bursts. Any combination of 10 Mb/s, 100 Mb/s, Half-Duplex, and Full-Duplex modes may be selected. The DP83840 will ad-

vertise all available abilities by default. Auto-Negotiation controls the exchange of configuration information. Upon successful Auto-Negotiation, the abilities reported by the Link Partner are stored in the Auto-Negotiation Link Partner Ability Register (ANLPAR) at address 05h.

The contents of the ANLPAR register are used to automatically configure to the highest performance protocol between the local and far-end nodes. Software can determine which mode has been configured by Auto-Negotiation by comparing the contents of the ANAR and ANLPAR registers and then selecting the technology whose bit is set in both the ANAR and ANLPAR of highest priority relative to the following list.

Auto-Negotiation Priority Resolution:

1. 100BASE-TX Full Duplex (Highest Priority)
2. 100BASE-T4
3. 100BASE-TX Half Duplex
4. 10BASE-T Full Duplex
5. 10BASE-T Half Duplex (Lowest Priority)

The Basic Mode Control Register (BCMR) at address 00h provides control of enabling, disabling, and restarting of the Auto-Negotiation function. When Auto-Negotiation is disabled the Speed Selection bit in the BCMR (bit 13, register address 00h) controls switching between 10 Mb/s or 100 Mb/s operation, while the Duplex Mode bit (bit 8, register address 00h) controls switching between full duplex operation and half duplex operation. The Speed Selection and Duplex Mode bits have no effect on the mode of operation when the Auto-Negotiation Enable bit (bit 12, register address 00h) is set.

The Basic Mode Status Register (BMSR) at address 01h indicates the set of available abilities for technology types (bits 15 to 11, register address 01h), Auto-Negotiation ability (bit 3, register address 01h), and Extended Register Capability (bit 0, register address 01h). These bits are permanently set to indicate the full functionality of the DP83840 (only the 100BASE-T4 bit is not set since the DP83840 does not support that function, while it does support all the other functions).

The BMSR also provides status on:

1. Whether Auto-Negotiation is complete (bit 5, register address 01h)
2. Whether the Link Partner is advertising that a remote fault has occurred (bit 4, register address 01h)
3. Whether a valid link has been established (bit 2, register address 01h)

The Auto-Negotiation Advertisement Register (ANAR) at address 04h indicates the Auto-Negotiation abilities to be advertised by the DP83840. All available abilities are transmitted by default, but any ability can be suppressed by writing to the ANAR. Updating the ANAR to suppress an ability is one way for a management agent to change (force) the technology that is used.

The Auto-Negotiation Link Partner Ability Register (ANLPAR) at address 05h indicates the abilities of the Link Partner as indicated by Auto-Negotiation communication. The contents of this register are considered valid when the Auto-Negotiation Complete bit (bit 5, register address 01h) is set.

### 3.0 Functional Description (Continued)

TABLE III. Auto-Negotiation Mode Select

| AN1<br>(Pin 46)         | AN0<br>(Pin 95) | Action                                                                                       | Mode                                                                                                   |
|-------------------------|-----------------|----------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------|
| <b>FORCED MODES</b>     |                 |                                                                                              |                                                                                                        |
| 0                       | M               | BMCR (00h) Bit 12 = 0, Bit 13 = 0, Bit 8 = 0<br>ANAR (04h) Bits 5 to 8 Set to Default Values | Auto-Negotiation Disabled with Only Half-Duplex<br>10BASE-T Forced                                     |
| 1                       | M               | BMCR (00h) Bit 12 = 0, Bit 13 = 0, Bit 8 = 1<br>ANAR (04h) Bits 5 to 8 Set to Default Values | Auto-Negotiation disabled with Only Full-Duplex<br>10BASE-T Forced                                     |
| M                       | 0               | BMCR (00h) Bit 12 = 0, Bit 13 = 1, Bit 8 = 0<br>ANAR (04h) Bits 5 to 8 Set to Default Values | Auto-Negotiation disabled with Only Half-Duplex<br>100BASE-X Forced                                    |
| M                       | 1               | BMCR (00h) Bit 12 = 0, Bit 13 = 1, Bit 8 = 1<br>ANAR (04h) Bits 5 to 8 Set to Default Values | Auto-Negotiation disabled with Only Full-Duplex<br>100BASE-X Forced                                    |
| <b>ADVERTISED MODES</b> |                 |                                                                                              |                                                                                                        |
| M                       | M               | BMCR (00h) Bit 12 = 1<br>ANAR (04h) Bits 5 to 8 = 1                                          | Auto-Negotiation Enabled for All Possible Protocols<br>(Half or Full Duplex for 10BASE-T or 100BASE-X) |
| 0                       | 0               | BMCR (00h) Bit 12 = 1<br>ANAR (04h) Bit 5 = 1, Bits 6 to 8 = 0                               | Auto-Negotiation Enabled with Only Half-Duplex<br>10BASE-T Available                                   |
| 0                       | 1               | BMCR (00h) Bit 12 = 1<br>ANAR (04h) Bit 5 = 0, Bit 6 = 1, Bits 7 and 8 = 0                   | Auto-Negotiation Enabled with Only Full-Duplex<br>10BASE-T Available                                   |
| 1                       | 0               | BMCR (00h) Bit 12 = 1<br>ANAR (04h) Bit 5 and 6 = 0, Bit 7 = 1, Bit 8 = 0                    | Auto-Negotiation Enabled with Only Half-Duplex<br>100BASE-X Available                                  |
| 1                       | 1               | BMCR (00h) Bit 12 = 1<br>ANAR (04h) Bit 5 to 7 = 0, Bit 8 = 1                                | Auto-Negotiation Enabled with Only Full-Duplex<br>100BASE-X Available                                  |

"M" indicates logic mid level ( $V_{CC} \div 2$ ) "1" indicates logic high level "0" indicates logic low level

The Auto-Negotiation Expansion Register (ANER) at address 06h indicates additional Auto-Negotiation status. The ANER provides status on:

- Whether a Multiple Link Fault has occurred (bit 4, register address 06h)
- Whether the Link Partner supports the Next Page function (bit 3, register address 06h)
- Whether the DP83840 supports the Next Page function (bit 2, register address 06h). The DP83840 does not support the Next Page function.
- Whether the current page being exchanged by Auto-Negotiation has been received (bit 1, register address 06h)
- Whether the Link Partner supports Auto-Negotiation (bit 0, register address 06h)

#### 3.9.3 Auto-Negotiation Parallel Detection

The DP83840 in conjunction with the DP83223 transceiver supports the Parallel Detection function as defined in the

IEEE 802.3u specification. Parallel Detection requires both the 10 Mb/s and 100 Mb/s receivers to monitor the receive signal and report link status to the Auto-Negotiation function. Auto-Negotiation uses this information to configure the correct technology in the event that the Link Partner does not support Auto-Negotiation, yet is transmitting link signals that the 100BASE-TX or 10BASE-T PMAs recognize as valid link signals. In the event that more than one PMA indicates a valid link, the Multiple Link Fault bit (bit 4, register address 06h) will be set.

As an example, when the Link Partner supports 100BASE-TX but does not support Auto-Negotiation, Parallel Detection will allow the DP83840 to negotiate to 100 Mb/s operation by detecting a valid set of IDLEs even though no Link Code Words were exchanged through FLP Bursts.

## 3.0 Functional Description (Continued)

### 3.10 RESET OPERATION

The DP83840 can be reset either by hardware or software. A hardware reset may be accomplished either by asserting the RESET pin (pin 44) during normal operation, or upon powering up the device. A software reset is accomplished by setting the reset bit in the Basic Mode Control Register (bit 15, address 00h).

#### 3.10.1 Power-Up Reset

When  $V_{CC}$  is first applied to the DP83840 it takes some amount of time for power to actually reach the nominal 5V potential. This initial power-up time can be referred to as a  $V_{CC}$  ramp when  $V_{CC}$  is "ramping" from 0V to 5V. When the initial  $V_{CC}$  ramp reaches approximately 4V, the DP83840 begins an internal reset operation which must be allowed sufficient time, relative to the assertion and deassertion of the RESET pin, to reset the device. There are two methods for guaranteeing successful reset upon device power-up.

The first method accounts for those designs that utilize a special power up circuit which, through hardware, will assert the RESET pin upon power-up. In this case, the deassertion (falling edge) of the RESET pin must not occur until at least 500  $\mu$ s after the time at which the  $V_{CC}$  ramp initially reached the 4V point.

The second method accounts for those applications which produce a reset pulse sometime after the initial power-up of the device. In this case, it is recommended that a positive pulse, with a duration of at least 1  $\mu$ s, be applied to the RESET pin no sooner than 500  $\mu$ s after the point in time where the initial  $V_{CC}$  ramp reached 4V.

In both methods described above, it is important to note that the logic levels present at each of the hardware configuration pins of the DP83840 (see list below) are also latched into the device as a function of the reset operation. These hardware configuration values are guaranteed to be latched into the DP83840 2  $\mu$ s after the deassertion of the RESET pin.

The hardware configuration values latched into the DP83840 during the reset operation are dependent on the logic levels present at the following device pins upon power-up:

| Pin # | Primary Function | Latched in at Reset |
|-------|------------------|---------------------|
| 49    | LBEN             | PHYAD[0]            |
| 53    | ENCSEL           | PHYAD[1]            |
| 66    | CRS              | PHYAD[2]            |
| 89    | PHYAD[3]         | PHYAD[3]            |
| 63    | RX_ER            | PHYAD[4]            |
| 95    | ANO              | ANO                 |
| 46    | AN1              | AN1                 |
| 47    | REPEATER         | REPEATER            |
| 98    | 10BTSER          | 10BTSER             |
| 99    | BPALIGN          | BPALIGN             |
| 100   | BP4B5B           | BP4B5B              |
| 1     | BPSCR            | BPSCR               |

During the power-up reset operation the  $\overline{\text{LED1}}$  through  $\overline{\text{LED5}}$  pins are undefined and the SPEED\_10 pin will be asserted.

#### 3.10.2 Hardware Reset

A hardware Reset is accomplished by applying a positive pulse, with a duration of at least 1  $\mu$ s, to the RESET pin of the DP83840 during normal operation. This will reset the device such that all registers will be reset to default values and the hardware configuration values will be re-latched into the device (similar to the Power-Up reset operation).

#### 3.10.3 Software Reset

A software reset is accomplished by setting bit 15 of the Basic Mode Control Register (address 00h). This bit is self clearing and, when set, will return a value of "1" until the software reset operation has completed. The software reset will only force the register bits with a given default state to that default state. Hardware configuration values will not be latched into the device as a result of a software reset.

### 3.11 LOOPBACK OPERATION

The DP83840 supports several different modes of loopback operation for diagnostic purposes.

#### 3.11.1 10BASE-T Loopback

The loopback options for 10BASE-T operation can be selected either by asserting the Loopback bit (bit 14) in the Basic Mode Control Register (address 00h), or by asserting the 10BT\_LPBK bit (bit 11) in the Loopback, Bypass and Receiver Error Mask Register (address 18h). Asserting either of these bits will cause the 10BASE-T data present at the transmit MII data inputs to be routed back to the receive MII data outputs. During this loopback mode, the Manchester encoded 10BASE-T data will not be present at either the TXU+/- or TXS+/- serial differential outputs.

Normal 10BASE-T operation, in order to be standard compliant, also loops back the MII transmit data to the MII receive data. However, the data is also allowed to be passed through the 10BASE-T transmitter and out either the TXU+/- or TXS+/- outputs as well.

#### 3.11.2 100BASE-X Loopback

The loopback options for 100BASE-X operation can be selected by asserting the Loopback bit (bit 14) in the Basic Mode Control Register (address 00h), or by selecting the desired mode as determined by the LB[1:0] (bits 8 and 9) in the Loopback, Bypass and Receiver Error Mask Register (address 18h).

Asserting the Loopback bit (bit 14) in the Basic Mode Control Register (address 00h) will cause the same loopback of MII transmit to MII receive as described previously in the 10BASE-T loopback section, except at 25 MHz due to 100BASE-X operation.

The LB[1:0] bits (bits 8 and 9) of the LBREMR (address 18h) allow for three different modes of operation:

- bit 8 = 0, bit 9 = 0; Normal operation without loopback
- bit 8 = 0, bit 9 = 1; PMD loopback operation
- bit 8 = 1, bit 9 = 0; Remote Loopback



### 3.0 Functional Description (Continued)

The first mode allows normal operation without any form of loopback.

The second mode asserts the LBEN output of the DP83840 which, when connected to the LBEN input of the twisted pair transceiver (DP83223A), forces the twisted pair transceiver into loopback mode. Therefore, when the DP83840 is transmitting 100BASE-X serial data from its serial TD+/- outputs to the twisted pair transceiver, this data is immediately routed back to the RD+/- 100BASE-X serial inputs of the DP83840 device.

The third mode selects the Remote Loopback operation. In this mode, the DP83840 device serves as a "remote loopback" for the far end partner. Serial data received off the twisted pair cable is routed, via the DP83223A, into the RD+/- serial inputs of the DP83840 where it is then routed back to the TD+/- serial outputs of the DP83840 and finally launched back onto the twisted pair cable, via the DP83223A, and sent back to the far-end partner.

In each of the 100BASE-X loopback modes, except for Remote Loopback, the assertion of the loopback function will result in a 330  $\mu$ s down-time where the 100BASE-TX descrambler must reacquire synchronization with the scrambled data stream before any valid data will appear at the receive MII RXD[3:0] outputs.

#### 3.12 ALTERNATIVE 100BASE-X OPERATION

The DP83840 10/100 Physical Layer device supports one standard and three alternative modes when operating at 100 Mb/s.

##### 3.12.1 Translational (normal) Mode

The first mode is referred to as the "Translational" mode. This is the standard and most commonly used operating mode where all transmit and receive functions are enabled in order to condition the data as it flows through the Physical Layer between the MAC and cable. All of the transmit and receive blocks as depicted in *Figures 4 and 5* are enabled (not bypassed).

##### 3.12.2 Transparent Mode

The second mode is referred to as "Transparent". In this mode, the 4B/5B translators in both the transmit and receive sections are bypassed as might be required in certain repeater applications. This is accomplished either by configuring the BP4B5B pin (100) of the DP83840 to a logic high level prior to power-up/hardware reset or by setting the BP\_4B5B bit (bit 14) of the LBREMR register (address 18h).

In "Transparent" mode, all remaining functional blocks within the 100BASE-X transmit and receive sections are still operational. This allows the 5B serial symbol on the twisted pair to be presented as descrambled data, without conversion to 4B, to the MII. Since the MII normally only carries a nibble wide word, the fifth bit, which is the new MSB, is carried on the RX\_ER and TX\_ER signals for receive and transmit operations respectively.

In the "Transparent" mode, all of the clock to data timing for both MII transmit and MII receive operations remains the same as in "Translational" mode. However, upon reception of a packet, the /J/K/ start of stream delimiter is not replaced by the /5/5/ MAC preamble nor is the /T/R/ end of stream delimiter removed from the packet before presentation to the MII receive RXD[3:0] and RX\_ER outputs.

Similarly, the transmit MII data TXD[3:0] and TX\_ER must already have /J/K/ and /T/R/ packet delimiters in place. Therefore, the repeater controller device is responsible for receiving the packet delimiters intact as well as transmitting these delimiters intact back to the DP83840 device(s).

The receive data valid flag, RX\_DV, operates the same during "Transparent" mode as it does in "Translational" mode. Additionally, Idles are passed to and from the MII as /00000/.

Finally, the "Transparent" mode of operation will operate the same when the DP83840 is in either node mode or repeater mode with the only difference being CRS functionality. As in "translational" mode, if the DP83840 is configured for repeater operation, the CRS signal will be suppressed during transmit such that only actual network collisions will be flagged.

##### 3.12.3 Phaser Mode

The final mode of operation at 100 Mb/s is referred to as the "Phaser" mode. This mode might be used for those applications where the system design requires only the clock recovery and clock generation functions of the DP83840. This is accomplished either by configuring the BPALIGN pin (99) of the DP83840 to a logic high level prior to power-up/hardware reset or by setting the BP\_ALIGN bit (bit 12) of the LBREMR register (address 18h).

In "Phaser" mode, all of the conditioning blocks in the transmit and receive sections of the 100BASE-X section are bypassed (refer to *Figures 4 and 5*). Therefore, whatever 5B data is presented to the MII transmit inputs (TXD[3:0] and TX\_ER) of the DP83840 is simply serialized and output to the DP83223A twisted pair transceiver to be sent out over the twisted pair cable. Similarly, the 100BASE-X serial data received at the RD+/- inputs of the DP83840 are shifted into 5-bit parallel words and presented to the MII receive outputs RXD[3:0] and RX\_ER. All data, including Idles, passes through the DP83840 unaltered other than for serial/parallel conversions.

In the "Phaser" mode, all of the clock to data timing for both MII transmit and MII receive operations remains the same as in "Translational" mode. Additionally, the "Phaser" mode will operate the same when the DP83840 is in either node mode or repeater mode with the only difference being CRS functionality. As in "translational" mode, if the DP83840 is configured for repeater operation, the CRS signal will be suppressed during transmit such that only actual network collisions will be flagged.

##### 3.12.4 100BASE-FX Mode

The DP83840 will allow 100BASE-FX functionality by bypassing the scrambler and descrambler. This can be accomplished either through hardware configuration or via software.

The hardware configuration is set simply by tying the BPSCR pin (1) high with a 4.7 k $\Omega$  resistor and then cycling power or resetting the DP83840. The software setting is accomplished by setting the BP\_SCR bit (bit 13) of the LBREMR register (address 18h) via MII serial management. It is important to bypass the disconnect function during 100BASE-FX operation by setting the F\_CONNECT bit (bit 5) in the PGR register (address 17h).

## 4.0 Registers

The MII supports up to 32 word-wide registers per addressable connected device. The DP83840's register allocation is as shown below. Each register is described in the Sections 4.2 to 4.17 that follow. Section 3.2 describes the MII serial access control method.

| Address | Register Name | Description                                                  |
|---------|---------------|--------------------------------------------------------------|
| 00h     | BMCR          | Basic Mode Control Register                                  |
| 01h     | BMSR          | Basic Mode Status Register                                   |
| 02h     | PHYIDR1       | PHY Identifier Register # 1                                  |
| 03h     | PHYIDR2       | PHY Identifier Register # 2                                  |
| 04h     | ANAR          | Auto-Negotiation Advertisement Register                      |
| 05h     | ANLPAR        | Auto-Negotiation Link Partner Ability Register               |
| 06h     | ANER          | Auto-Negotiation Expansion Register                          |
| 7h–Fh   | Reserved      | Reserved for Future Assignments by the MII Working Group     |
| 10h–11h | Reserved      | Reserved for PHY Specific Future Assignments by Vendor       |
| 12h     | DCR           | Disconnect Counter Register                                  |
| 13h     | FCSCR         | False Carrier Sense Counter Register                         |
| 14h     | Reserved      | Reserved—Do Not Read/Write to this Register                  |
| 15h     | RECR          | Receive Error Counter Register                               |
| 16h     | SRR           | Silicon Revision Register                                    |
| 17h     | PCR           | PCS Sub-Layer Configuration Register                         |
| 18h     | LBREMR        | Loopback, Bypass and Receiver Error Mask Register            |
| 19h     | PAR           | PHY Address Register                                         |
| 1Ah     | Reserved      | Reserved for PHY Specific Future Assignment by Vendor        |
| 1Bh     | 10BTSR        | 10BASE-T Status Register                                     |
| 1Ch     | 10BTCR        | 10BASE-T Configuration Register                              |
| 1Dh–1Fh | Reserved      | Reserved for Future Use—Do Not Read/Write to These Registers |

### 4.1 KEY TO DEFAULTS

In the register descriptions that follow, the default column takes the form:

<reset value>, <access type> / <attribute(s)>

Where:

|                |                                      |                           |
|----------------|--------------------------------------|---------------------------|
| <reset value>: |                                      | <access type>:            |
| 1              | Bit Set to Logic One                 | RO = Read Only            |
| 0              | Bit Set to Logic Zero                | RW = Read/Write           |
| X              | No Default Value                     | <attribute(s)>:           |
| (Pin #)        | Value Latched in from Pin # at Reset | L = Latching              |
|                |                                      | SC = Self Clearing        |
|                |                                      | P = Value Permanently Set |

## 4.0 Registers (Continued)

### 4.2 BASIC MODE CONTROL REGISTER (BMCR)

Address 00h

| Bit | Bit Name                 | Default                | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----|--------------------------|------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15  | Reset                    | 0, RW/SC               | <p><b>RESET:</b></p> <p>1 = Software Reset<br/>0 = Normal Operation</p> <p>This bit sets the status and control registers of the PHY to their default states. This bit, which is self-clearing, returns a value of one until the reset process is complete.</p>                                                                                                                                                                                                                                                                              |
| 14  | Loopback                 | 0, RW                  | <p><b>LOOPBACK:</b></p> <p>1 = Loopback Enabled<br/>0 = Normal Operation</p> <p>The loopback function enables MII transmit data to be routed to the MII receive data path.</p> <p>Setting this bit may cause the descrambler to lose synchronization and produce a 330 <math>\mu</math>s "dead time" before any valid data will appear at the MII receive outputs.</p> <p>This bit takes priority over the loopback control bits 8 and 9 in the LBREMR register (address 18h).</p>                                                           |
| 13  | Speed Selection          | 1, RW                  | <p><b>SPEED SELECT:</b></p> <p>1 = 100 Mb/s<br/>0 = 10 Mb/s</p> <p>Link speed is selected by this bit or by Auto-Negotiation if bit 12 of this register is set (in which case, the value of this bit is ignored).</p>                                                                                                                                                                                                                                                                                                                        |
| 12  | Auto-Negotiation Enable  | 1, RW                  | <p><b>AUTO-NEGOTIATION ENABLE:</b></p> <p>1 = Auto-Negotiation Enabled—bits 8 and 13 of this register are ignored when this bit is set.<br/>0 = Auto-Negotiation Disabled—bits 8 and 13 determine the link speed and mode.</p>                                                                                                                                                                                                                                                                                                               |
| 11  | Power Down               | 0, RW                  | <p><b>POWER DOWN:</b></p> <p>1 = Power Down<br/>0 = Normal Operation</p> <p>The power-down mode is not currently implemented. Setting this bit has no effect.</p>                                                                                                                                                                                                                                                                                                                                                                            |
| 10  | Isolate                  | (PHYAD = 00000),<br>RW | <p><b>ISOLATE:</b></p> <p>1 = Isolates the DP83840 from the MII with the exception of the serial management. When this bit is asserted, the DP83840 does not respond to TXD[3:0], TX_EN, and TX_ER inputs, and it presents a high impedance on its TX_CLK, RX_CLK, RX_DV, RX_ER, RXD[3:0], COL and CRS outputs. The CLK_25M output stays active and the DP83840 still responds to serial management transactions.</p> <p>0 = Normal Operation</p> <p>If the PHY Address is set to 00000 the Isolate bit will be set upon power-up/reset.</p> |
| 9   | Restart Auto-Negotiation | 0, RW/SC               | <p><b>RESTART AUTO-NEGOTIATION:</b></p> <p>1 = Restart Auto-Negotiation. Re-initiates the Auto-Negotiation process. If Auto-Negotiation is disabled (bit 12 of this register cleared), this bit has no function and should be cleared. This bit is self-clearing and will return a value of 1 until Auto-Negotiation is initiated by the DP83840, whereupon it will self-clear. Operation of the Auto-Negotiation process is not affected by the management entity clearing this bit.</p> <p>0 = Normal Operation</p>                        |

## 4.0 Registers (Continued)

### 4.2 BASIC MODE CONTROL REGISTER (BMCR) (Continued)

| Bit | Bit Name       | Default | Description                                                                                                                                                                                                                                                                                                                |
|-----|----------------|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 8   | Duplex Mode    | 1, RW   | <b>DUPLEX MODE:</b><br>1 = Full Duplex operation. Duplex selection is allowed when Auto-Negotiation is disabled (bit 12 of this register is cleared). With Auto-Negotiation enabled, this bit reflects the duplex capability as specified in bits 11 to 15 of the BMSR register (address 1h).<br>0 = Half Duplex Operation |
| 7   | Collision Test | 0, RW   | <b>COLLISION TEST:</b><br>1 = Collision Test enabled. When set, this bit will cause the COL signal to be asserted in response to the assertion of TX_ER.<br>0 = Normal Operation                                                                                                                                           |
| 6:0 | Reserved       | X, RO   | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                                                           |

### 4.3 BASIC MODE STATUS REGISTER (BMSR)

Address 01h

| Bit  | Bit Name                  | Default | Description                                                                                                                                                                                                                                                                                            |
|------|---------------------------|---------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15   | 100BASE-T4                | 0, RO/P | <b>100BASE-T4 CAPABLE:</b><br>1 = DP83840 able to perform in 100BASE-T4 mode<br>0 = DP83840 not able to perform in 100BASE-T4 mode                                                                                                                                                                     |
| 14   | 100BASE-TX Full Duplex    | 1, RO/P | <b>100BASE-TX FULL DUPLEX CAPABLE:</b><br>1 = DP83840 able to perform 100BASE-TX in full duplex mode<br>0 = DP83840 not able to perform 100BASE-TX in full duplex mode                                                                                                                                 |
| 13   | 100BASE-TX Half Duplex    | 1, RO/P | <b>100BASE-TX HALF DUPLEX CAPABLE:</b><br>1 = DP83840 able to perform 100BASE-TX in half duplex mode<br>0 = DP83840 not able to perform 100BASE-TX in half duplex mode                                                                                                                                 |
| 12   | 10BASE-T Full Duplex      | 1, RO/P | <b>10BASE-T FULL DUPLEX CAPABLE:</b><br>1 = DP83840 able to perform 10BASE-T in full duplex mode<br>0 = DP83840 not able to perform 10BASE-T in full duplex mode                                                                                                                                       |
| 11   | 10BASE-T Half Duplex      | 1, RO/P | <b>10BASE-T HALF DUPLEX CAPABLE:</b><br>1 = DP83840 able to perform 10BASE-T in half duplex mode<br>0 = DP83840 not able to perform 10BASE-T in half duplex mode                                                                                                                                       |
| 10:6 | Reserved                  | 0, RO   | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                                       |
| 5    | Auto-Negotiation Complete | 0, RO   | <b>AUTO-NEGOTIATION COMPLETE:</b><br>1 = Auto-Negotiation process complete<br>0 = Auto-Negotiation process not complete                                                                                                                                                                                |
| 4    | Remote Fault              | 0, RO/L | <b>REMOTE FAULT:</b><br>1 = Remote Fault condition detected (cleared on read or by a chip reset). Fault criteria and detection method is DP83840 implementation specific. This bit is set if the RF bit in the ANLPAR (bit 13, register address 05h) is set.<br>0 = No remote fault condition detected |
| 3    | Auto-Negotiation Ability  | 1, RO/P | <b>AUTO CONFIGURATION ABILITY:</b><br>1 = DP83840 is able to perform Auto-Negotiation<br>0 = DP83840 is not able to perform Auto-Negotiation                                                                                                                                                           |

## 4.0 Registers (Continued)

### 4.3 BASIC MODE STATUS REGISTER (BMSR) (Continued)

Address 01h

| Bit | Bit Name            | Default | Description                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-----|---------------------|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 2   | Link Status         | 0, RO/L | <p><b>LINK STATUS:</b></p> <p>1 = Valid link established (for either 10 Mb/s or 100 Mb/s operation)<br/>0 = Link not established</p> <p>The criteria for link validity is implementation specific. The link status bit is implemented with a latching function, so that the occurrence of a link failure condition causes the Link Status bit to become cleared and remain cleared until it is read via the management interface.</p> |
| 1   | Jabber Detect       | 0, RO/L | <p><b>JABBER DETECT:</b></p> <p>1 = Jabber condition detected<br/>0 = No Jabber</p> <p>This bit is implemented with a latching function so that the occurrence of a jabber condition causes it to become set until it is cleared by a read to this register by the management interface or by a DP83840 reset. This bit only has meaning in 10 Mb/s mode.</p>                                                                         |
| 0   | Extended Capability | 1, RO/P | <p><b>EXTENDED CAPABILITY:</b></p> <p>1 = Extended register capable<br/>0 = Basic register capable only</p>                                                                                                                                                                                                                                                                                                                           |

### 4.4 PHY IDENTIFIER REGISTER #1 (PHYIDR1)

Address 02h

The PHY Identifier Registers #1 and #2 together form a unique identifier for the DP83840. The Identifier consists of a concatenation of the Organizationally Unique Identifier (OUI), the vendor's model number and the model revision number. A PHY may return a value of zero in each of the 32 bits of the PHY Identifier if desired. The PHY Identifier is intended to support network management.

National Semiconductor's IEEE assigned OUI is 080017h.

| Bit  | Bit Name | Default                            | Description                                                                                                                                                                                                                                              |
|------|----------|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15:0 | OUI__MSB | <00 1000<br>0000 0000<br>00>, RO/P | <p><b>OUI MOST SIGNIFICANT BITS:</b> This register stores bits 3 to 18 of the OUI (080017h) to bits 15 to 0 of this register respectively. The most significant two bits of the OUI are ignored (the IEEE standard refers to these as bits 1 and 2).</p> |

### 4.5 PHY IDENTIFIER REGISTER #2 (PHYIDR2)

Address 03h

| Bit   | Bit Name  | Default            | Description                                                                                                                                                                                                |
|-------|-----------|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15:10 | OUI__LSB  | <01 0111>,<br>RO/P | <p><b>OUI LEAST SIGNIFICANT BITS:</b> Bits 19 to 24 of the OUI (080017h) are mapped to bits 15 to 10 of this register respectively.</p>                                                                    |
| 9:4   | VNDR__MDL | <00 0000>,<br>RO/P | <p><b>VENDOR MODEL NUMBER:</b> Six bits of vendor model number mapped to bits 9 to 4 (most significant bit to bit 9).</p>                                                                                  |
| 3:0   | MDL__REV  | <0000>,<br>RO/P    | <p><b>MODEL REVISION NUMBER:</b> Four bits of vendor model revision number mapped to bits 3 to 0 (most significant bit to bit 3). This field will be incremented for all major DP83840 device changes.</p> |

## 4.0 Registers (Continued)

### 4.6 AUTO-NEGOTIATION ADVERTISEMENT REGISTER (ANAR)

Address 04h

This register contains the advertised abilities of this DP83840 device as they will be transmitted to its Link Partner during Auto-Negotiation.

| Bit   | Bit Name | Default     | Description                                                                                                                                                                                                                                                                                                                                      |
|-------|----------|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15    | NP       | 0, RO/P     | <b>NEXT PAGE INDICATION:</b><br>0 = Not Next Page able<br>1 = Next Page able<br>The DP83840 is not Next Page capable so this bit is permanently set to 0.                                                                                                                                                                                        |
| 14    | ACK      | 0, RO/P     | <b>ACKNOWLEDGE:</b><br>1 = Reception of Link Partner ability data acknowledged<br>0 = Not acknowledged<br>The DP83840's Auto-Negotiation state machine will automatically control this bit in the outgoing FLP bursts, setting it at the appropriate time during the Auto-Negotiation process. Software should not attempt to write to this bit. |
| 13    | RF       | 0, RW       | <b>REMOTE FAULT:</b><br>1 = Advertises that this device has detected a Remote Fault<br>0 = No Remote Fault detected                                                                                                                                                                                                                              |
| 12:10 | Reserved | X, RW       | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                                                                                 |
| 9     | T4       | 0, RO/P     | <b>100BASE-T4 SUPPORT:</b><br>1 = 100BASE-T4 is supported by the local device<br>0 = 100BASE-T4 not supported<br>The DP83840 does not support 100BASE-T4 so this bit is permanently set to 0.                                                                                                                                                    |
| 8     | TX_FD    | 1, RW       | <b>100BASE-TX FULL DUPLEX SUPPORT:</b><br>1 = 100BASE-TX Full Duplex is supported by the local device<br>0 = 100BASE-TX Full Duplex not supported                                                                                                                                                                                                |
| 7     | TX       | 1, RW       | <b>100BASE-TX SUPPORT:</b><br>1 = 100BASE-TX is supported by the local device<br>0 = 100BASE-TX not supported                                                                                                                                                                                                                                    |
| 6     | 10_FD    | 1, RW       | <b>10BASE-T FULL DUPLEX SUPPORT:</b><br>1 = 10BASE-T Full Duplex is supported by the local device<br>0 = 10BASE-T Full Duplex not supported                                                                                                                                                                                                      |
| 5     | 10       | 1, RW       | <b>10BASE-T SUPPORT:</b><br>1 = 10BASE-T is supported by the local device<br>0 = 10BASE-T not supported                                                                                                                                                                                                                                          |
| 4:0   | Selector | <00001>, RW | <b>PROTOCOL SELECTION BITS:</b> These bits contain the binary encoded protocol selector supported by this node.<br><00001> indicates that this device supports IEEE 802.3 CSMA/CD                                                                                                                                                                |

## 4.0 Registers (Continued)

### 4.7 AUTO-NEGOTIATION LINK PARTNER ABILITY REGISTER (ANLPAR)

Address 05h

This register contains the advertised abilities of the Link Partner as received during Auto-Negotiation.

| Bit   | Bit Name | Default     | Description                                                                                                                                                                                                                                                                                        |
|-------|----------|-------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15    | NP       | 0, RO       | <b>NEXT PAGE INDICATION:</b><br>0 = Link Partner not Next Page able<br>1 = Link Partner is Next Page able                                                                                                                                                                                          |
| 14    | ACK      | 0, RO       | <b>ACKNOWLEDGE:</b><br>1 = Link Partner acknowledges reception of the ability data word<br>0 = Not acknowledged<br><br>The DP83840's Auto-Negotiation state machine will automatically control the use of this bit from the incoming FLP bursts. Software should not attempt to write to this bit. |
| 13    | RF       | 0, RO       | <b>REMOTE FAULT:</b><br>1 = Remote Fault indicated by Link Partner<br>0 = No Remote Fault indicated by Link Partner                                                                                                                                                                                |
| 12:10 | Reserved | X, RO       | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                                   |
| 9     | T4       | 0, RO       | <b>100BASE-T4 SUPPORT:</b><br>1 = 100BASE-T4 is supported by the Link Partner<br>0 = 100BASE-T4 not supported by the Link Partner                                                                                                                                                                  |
| 8     | TX_FD    | 0, RO       | <b>100BASE-TX FULL DUPLEX SUPPORT:</b><br>1 = 100BASE-TX Full Duplex is supported by the Link Partner<br>0 = 100BASE-TX Full Duplex not supported by the Link Partner                                                                                                                              |
| 7     | TX       | 0, RO       | <b>100BASE-TX SUPPORT:</b><br>1 = 100BASE-TX is supported by the Link Partner<br>0 = 100BASE-TX not supported by the Link Partner                                                                                                                                                                  |
| 6     | 10_FD    | 0, RO       | <b>10BASE-T FULL DUPLEX SUPPORT:</b><br>1 = 10BASE-T Full Duplex is supported by the Link Partner<br>0 = 10BASE-T Full Duplex not supported by the Link Partner                                                                                                                                    |
| 5     | 10       | 0, RO       | <b>10BASE-T SUPPORT:</b><br>1 = 10BASE-T is supported by the Link Partner<br>0 = 10BASE-T not supported by the Link Partner                                                                                                                                                                        |
| 4:0   | Selector | <00000>, RO | <b>PROTOCOL SELECTION BITS:</b> Link Partner's binary encoded protocol selector.                                                                                                                                                                                                                   |

## 4.0 Registers (Continued)

### 4.8 AUTO-NEGOTIATION EXPANSION REGISTER (ANER)

Address 06h

| Bit  | Bit Name   | Default | Description                                                                                                                                                                                                                                                                                                           |
|------|------------|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15:5 | Reserved   | 0, RO   | <b>RESERVED:</b> Always 0.                                                                                                                                                                                                                                                                                            |
| 4    | MLF        | 0, RO   | <b>MULTIPLE LINK FAULT:</b><br>1 = Multiple Link Fault—indicates that it was not possible to resolve the connection because the Link Partner did not support Auto-Negotiation or was unable to distinguish a common ability.<br>0 = No Multiple Link Fault<br>This Feature is not currently supported in the DP83840. |
| 3    | LP_NP_ABLE | 0, RO   | <b>LINK PARTNER NEXT PAGE ABLE:</b> Status indicating if the Link Partner supports Next Page negotiation. A one indicates that the Link Partner does support Next Page.<br>This Feature is not currently supported in the DP83840.                                                                                    |
| 2    | NP_ABLE    | 0, RO/P | <b>NEXT PAGE ABLE:</b> Indicates if this node is able to send additional "Next Pages". The DP83840 is not Next Page Able, so this bit is always zero.                                                                                                                                                                 |
| 1    | PAGE_RX    | 0, RO   | <b>LINK CODE WORD PAGE RECEIVED:</b> This bit is set when a new Link Code Word Page has been received. This bit is automatically cleared when the Auto-Negotiation Link Partner Ability Register (ANLPAR register 05h) is read by management.<br>This Feature is not currently supported in the DP83840.              |
| 0    | LP_AN_ABLE | 0, RO   | <b>LINK PARTNER AUTO-NEGOTIATION ABLE:</b> A one in this bit indicates that the Link Partner supports Auto-Negotiation.                                                                                                                                                                                               |

### 4.9 DISCONNECT COUNTER REGISTER (DCR)

Address 12h

| Bit  | Bit Name   | Default        | Description                                                                                                                                                                                                                                |
|------|------------|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15:0 | DCNT[15:0] | <0000h>, RW/SC | <b>DISCONNECT COUNTER:</b> This 16-bit counter increments for each disconnect event. Each time this DP83840 and its Link Partner are disconnected from each other, the counter increments. This counter automatically rolls over to 0000h. |

### 4.10 FALSE CARRIER SENSE COUNTER REGISTER (FCSCR)

Address 13h

| Bit  | Bit Name     | Default        | Description                                                                                                                                                                                                       |
|------|--------------|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15:0 | FCSCNT[15:0] | <0000h>, RW/SC | <b>FALSE CARRIER EVENT COUNTER:</b> This 16-bit counter increments for each false carrier event, that is, when carrier sense is asserted without J/K symbol detection. This counter freezes when full (at FFFFh). |



## 4.0 Registers (Continued)

### 4.11 RECEIVE ERROR COUNTER REGISTER (RECR)

Address 15h

| Bit  | Bit Name      | Default        | Description                                                                                                                                                                                                                                                                                                                                                |
|------|---------------|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15:0 | RXERCNT[15:0] | <0000h>, RW/SC | <b>RX_ER COUNTER:</b> This 16-bit counter is incremented for each packet in which a receive error is detected. If there are one or more receiver error conditions during a valid packet reception (i.e. no collision occurred during packet reception), the counter is incremented once at the end of packet reception. This counter rolls over when full. |

### 4.12 SILICON REVISION REGISTER (SRR)

Address 16h

| Bit  | Bit Name    | Default     | Description                                                                                                       |
|------|-------------|-------------|-------------------------------------------------------------------------------------------------------------------|
| 15:8 | SIREV[15:8] | <00h>, RO/P | <b>ARCHITECTURE LEVEL:</b> This number will be incremented at the next major architectural change to the device.  |
| 7:8  | SIREV[7:0]  | <00h>, RO/P | <b>CIRCUIT ENHANCEMENT LEVEL:</b> This number will be incremented at the next minor circuit change to the device. |

### 4.13 PCS CONFIGURATION REGISTER (PCR)

Address 17h

| Bit  | Bit Name  | Default       | Description                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|------|-----------|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15   | NRZI_EN   | 1, RW         | <b>NRZI ENABLE:</b><br>1 = NRZI encoding and decoding of the transmit and receive data streams<br>0 = NRZI encoding and decoding disabled                                                                                                                                                                                                                                                                                                                 |
| 14   | Reserved  | X, RO         | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                                                                                                                                                                                          |
| 13   | TO_DIS    | 0, RW         | <b>TIMEOUT DISABLE:</b><br>1 = Timeout Counter in the descrambler section of the receiver disabled<br>0 = Timeout Counter enabled                                                                                                                                                                                                                                                                                                                         |
| 12   | REPEATER  | (Pin #47), RW | <b>REPEATER/NODE MODE:</b><br>1 = Repeater mode<br>0 = Node mode<br><br>In repeater mode the Carrier Sense (CRS) output from the DP83840 is asserted due to receive activity only. In node mode, and not configured for Full Duplex operation, CRS is asserted due to either receive or transmit activity.<br><br>The value of the REPEATER pin 47 (set by a pull-up or pull-down resistor, typically 4.7 kΩ) is latched into this bit at power-up/reset. |
| 11   | ENCSEL    | 0, RW         | <b>ENCODER MODE SELECT:</b><br>1 = External transceiver binary encoding<br>0 = External transceiver MLT3 encoding<br><br>This bit drives the DP83840's ENCSEL signal (pin 53). ENCSEL should be connected to the ENCSEL input of a DP83223 Twister.                                                                                                                                                                                                       |
| 10:8 | Reserved  | X, RO         | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                                                                                                                                                                                          |
| 7    | CLK25MDIS | 0, RW         | <b>CLK25M DISABLE:</b><br>1 = CLK25M output clock signal (pin 81) tri-stated<br>0 = CLK25M enabled<br><br>This helps reduce ground bounce and power consumption should this output not be required. For applications requiring the CLK25M output, leave this bit set to 0. See Section 3.5 for more details.                                                                                                                                              |

## 4.0 Registers (Continued)

### 4.13 PCS CONFIGURATION REGISTER (PCR) (Continued)

Address 17h

| Bit | Bit Name   | Default | Description                                                                                                                                                                                                                                                                           |
|-----|------------|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 6   | F_LINK_100 | 1, RW   | <b>FORCE GOOD LINK IN 100 Mb/s:</b><br>1 = Normal 100 Mb/s operation<br>0 = Force 100 Mb/s Good Link status<br>This bit is useful for diagnostic purposes.                                                                                                                            |
| 5   | F_CONNECT  | 0, RW   | <b>FORCE DISCONNECT FUNCTION BYPASS:</b><br>1 = Force Link Disconnect Function to be bypassed<br>0 = Normal operation                                                                                                                                                                 |
| 4   | TX_OFF     | 0, RW   | <b>FORCE TRANSMIT OFF:</b><br>1 = TD+ and TD- 100 Mb/s outputs forced to be inactive<br>0 = Normal 100 Mb/s transmission enabled                                                                                                                                                      |
| 3   | Reserved   | X, RO   | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                      |
| 2   | LED1_MODE  | 0, RW   | <b>LED1 MODE SELECT:</b><br>1 = LED1 output (pin 42) configured to indicate connection status (CON_STATUS, bit 5 of the PAR, address 19h). This is useful for network management purposes in 100BASE-TX mode.<br>0 = Normal LED1 operation—10 Mb/s and 100 Mb/s transmission activity |
| 1   | LED4_MODE  | 0, RW   | <b>LED4 MODE SELECT:</b><br>1 = LED4 output (pin 37) configured to indicate Full Duplex mode status for 10 Mb/s and 100 Mb/s operation<br>0 = LED4 output configured to indicate Polarity in 10BASE-T mode or Full Duplex in 100BASE-TX mode                                          |
| 0   | Reserved   | X, RO   | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                      |

### 4.14 LOOPBACK, BYPASS AND RECEIVER ERROR MASK REGISTER (LBREMR)

Address 18h

| Bit | Bit Name  | Default            | Description                                                                                                                                                                                                                                                                                                |
|-----|-----------|--------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15  | Reserved  | 1, RW              | <b>RESERVED:</b> Write as 1, read as don't care.                                                                                                                                                                                                                                                           |
| 14  | BP_4B5B   | (Pin # 100),<br>RW | <b>BYPASS 4B5B ENCODING AND 5B4B DECODING:</b> The value of the BP4B5B pin (100) is latched into this bit at power-up/reset.<br>1 = 4B5B encoder and 5B4B decoder functions bypassed<br>0 = Normal 4B5B and 5B4B operation                                                                                 |
| 13  | BP_SCR    | (Pin # 1), RW      | <b>BYPASS SCRAMBLER/DESCRAMBLER FUNCTION:</b> The value of the BPSCR pin (1) is latched into this bit at power-up/reset.<br>1 = Scrambler and descrambler functions bypassed<br>0 = Normal scrambler and descrambler operation                                                                             |
| 12  | BP_ALIGN  | (Pin # 99),<br>RW  | <b>BYPASS SYMBOL ALIGNMENT FUNCTION:</b> The value of the BPALIGN pin (99) is latched into this bit at power-up/reset.<br>1 = Receive functions (descrambler, symbol alignment and symbol decoding functions) bypassed. Transmit functions (symbol encoder and scrambler) bypassed<br>0 = Normal operation |
| 11  | 10BT_LPBK | 0, RW              | <b>10BASE-T ENCODER/DECODER LOOPBACK:</b><br>1 = Data loopback in the 10BASE-T ENDEC enabled<br>0 = Normal Operation                                                                                                                                                                                       |
| 10  | Reserved  | (pin # 49),<br>RW  | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                                           |

## 4.0 Registers (Continued)

### 4.14 LOOPBACK, BYPASS AND RECEIVER ERROR MASK REGISTER (LBREMR) (Continued)

| Bit | Bit Name | Default                                                                                                                      | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |     |     |      |   |   |             |   |   |                          |   |   |                                                                                                                              |   |   |          |
|-----|----------|------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|-----|------|---|---|-------------|---|---|--------------------------|---|---|------------------------------------------------------------------------------------------------------------------------------|---|---|----------|
| 9:8 | LB[1:0]  | <00>, RW                                                                                                                     | <p><b>LOOPBACK CONTROL BITS 1:0</b> : These bits control the 100 Mb/s loopback function as follows:</p> <table border="0"> <thead> <tr> <th>LB1</th> <th>LB0</th> <th>Mode</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>0</td> <td>Normal Mode</td> </tr> <tr> <td>0</td> <td>1</td> <td>DP83223 Twister Loopback</td> </tr> <tr> <td>1</td> <td>0</td> <td>Remote Loopback—Received data is looped back to the transmit channel at the MII. This is useful for bit error rate testings.</td> </tr> <tr> <td>1</td> <td>1</td> <td>Reserved</td> </tr> </tbody> </table> <p>Note that Twister Loopback, like the internal loopback described in the BMCR bit 14 (address 00h), will produce a "dead time" of 330 <math>\mu</math>s before any valid data appears at the RXD outputs.</p> | LB1 | LB0 | Mode | 0 | 0 | Normal Mode | 0 | 1 | DP83223 Twister Loopback | 1 | 0 | Remote Loopback—Received data is looped back to the transmit channel at the MII. This is useful for bit error rate testings. | 1 | 1 | Reserved |
| LB1 | LB0      | Mode                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |     |     |      |   |   |             |   |   |                          |   |   |                                                                                                                              |   |   |          |
| 0   | 0        | Normal Mode                                                                                                                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |     |     |      |   |   |             |   |   |                          |   |   |                                                                                                                              |   |   |          |
| 0   | 1        | DP83223 Twister Loopback                                                                                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |     |     |      |   |   |             |   |   |                          |   |   |                                                                                                                              |   |   |          |
| 1   | 0        | Remote Loopback—Received data is looped back to the transmit channel at the MII. This is useful for bit error rate testings. |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |     |     |      |   |   |             |   |   |                          |   |   |                                                                                                                              |   |   |          |
| 1   | 1        | Reserved                                                                                                                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |     |     |      |   |   |             |   |   |                          |   |   |                                                                                                                              |   |   |          |
| 7:5 | Reserved | 0, RW                                                                                                                        | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |     |     |      |   |   |             |   |   |                          |   |   |                                                                                                                              |   |   |          |
| 4   | CODE_ERR | 0, RW                                                                                                                        | <p><b>CODE ERRORS:</b></p> <p>1 = Forces code errors to be reported with the value 5h on RXD[3:0] and with RX_ER set to 1</p> <p>0 = Forces code errors to be reported with the value 6h on RXD[3:0] and with RX_ER set to 1</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |     |     |      |   |   |             |   |   |                          |   |   |                                                                                                                              |   |   |          |
| 3   | PE_ERR   | 0, RW                                                                                                                        | <p><b>PREMATURE END ERRORS:</b></p> <p>1 = Forces premature end errors to be reported with the value 4h on RXD[3:0] and with RX_ER set to 1</p> <p>0 = Forces premature end errors to be reported with the value 6h on RXD[3:0] and with RX_ER set to 1</p> <p>Premature end errors are caused by the detection of two IDLE symbols in the receive data stream prior to the T/R symbol pair denoting end of stream delimiter.</p>                                                                                                                                                                                                                                                                                                                                                           |     |     |      |   |   |             |   |   |                          |   |   |                                                                                                                              |   |   |          |
| 2   | LINK_ERR | 0, RW                                                                                                                        | <p><b>LINK ERRORS:</b></p> <p>1 = Forces link errors to be reported with the value 3h on RXD[3:0] and with RX_ER set to 1</p> <p>0 = Data is passed to RXD[3:0] unchanged and with RX_ER set to 0</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |     |     |      |   |   |             |   |   |                          |   |   |                                                                                                                              |   |   |          |
| 1   | PKT_ERR  | 0, RW                                                                                                                        | <p><b>PACKET ERRORS:</b></p> <p>1 = Forces packet errors (722 <math>\mu</math>s timeout) to be reported with the value 2h on RXD[3:0] and with RX_ER set to 1</p> <p>0 = Data is passed to RXD[3:0] unchanged and with RX_ER set to 0</p>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |     |     |      |   |   |             |   |   |                          |   |   |                                                                                                                              |   |   |          |
| 0   | Reserved | 0, RW                                                                                                                        | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |     |     |      |   |   |             |   |   |                          |   |   |                                                                                                                              |   |   |          |

## 4.0 Registers (Continued)

### 4.15 PHY ADDRESS REGISTER (PAR)

Address 19h

| Bit  | Bit Name     | Default        | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|------|--------------|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15:7 | Reserved     | 0, RO          | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| 6    | SPEED_10     | RO             | <p><b>SPEED INDICATION:</b> This bit indicates the current operational speed of the DP83840.</p> <p>1 = 10 Mb/s operation<br/>0 = 100 Mb/s operation</p> <p>This bit is only valid if a good link condition has been met.<br/>Good link is indicated when bit 2 of the BMSR (address 01h) is set.</p>                                                                                                                                                                                                     |
| 5    | CON_STATUS   | 0, RO/L        | <p><b>CONNECT STATUS:</b> This bit indicates the status of the disconnect function. The connect status is optionally muxed out through the LED1 pin when the LED1_MODE register bit (bit 2 of the PCR, address 17h) is asserted.</p> <p>1 = Valid link connection detected<br/>0 = Valid link connection not detected</p>                                                                                                                                                                                 |
| 4:0  | PHYADDR[4:0] | (PHYAD),<br>RW | <p><b>PHY ADDRESS BITS 4:0:</b> The values of the PHYAD[4:0] pins are latched to this register at power-up/reset. See Section 2.8 for the description of these pins.</p> <p>The first PHY address bit transmitted or received is the MSB of the address (bit 4). A station management entity connected to multiple PHY entities must know the appropriate address of each PHY.</p> <p>A PHY address of &lt;00000&gt; will cause the Isolate bit of the BMCR (bit 10, register address 00h) to be set.</p> |

### 4.16 10BASE-T STATUS REGISTER (10BTSR)

Address 1Bh

| Bit   | Bit Name | Default         | Description                                                                                                                                                                                                                                                                                                                        |
|-------|----------|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15:10 | Reserved | 0, RO           | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                                                                   |
| 9     | 10BT_SER | (Pin#98),<br>RW | <p><b>10BASE-T SERIAL MODE:</b> The value on the 10BTSER pin (98) is latched into this bit at power-up/reset.</p> <p>1 = 10BASE-T serial mode selected (see Sections 2.5 and 3.1.3.3 for more details)<br/>0 = 10BASE-T nibble mode selected (see Section 3.1.3.2)</p> <p>Serial mode is not supported for 100 Mb/s operation.</p> |
| 8:0   | Reserved | 0, RO           | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                                                                   |

## 4.0 Registers (Continued)

### 4.17 10BASE-T CONFIGURATION REGISTER (10BTCCR)

Address 1Ch

| Bit  | Bit Name | Default | Description                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|------|----------|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 15:6 | Reserved | 0, RO/P | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                                                                                                                                                                                      |
| 5    | LP_EN    | 1, RW   | <p><b>LINK PULSE ENABLE:</b></p> <p>1 = Transmission of link pulses enabled<br/>0 = Link pulses disabled, good link condition forced</p> <p>When configured for 100 Mb/s operation with Auto-Negotiation enabled, clearing this bit will force the DP83840 into 10 Mb/s operation with link pulses disabled.</p> <p>If the DP83840 has been configured for 100 Mb/s operation with Auto-Negotiation disabled, this bit will not affect operation.</p> |
| 4    | HBE      | 1, RW   | <p><b>HEARTBEAT ENABLE:</b></p> <p>1 = Heartbeat function enabled<br/>0 = Heartbeat function disabled</p> <p>When the DP83840 is configured for Full Duplex operation, this bit will be ignored (the collision/heartbeat function has no meaning in Full Duplex mode).</p>                                                                                                                                                                            |
| 3    | UTP/STP  | 1, RW   | <p><b>UTP/STP MEDIA SELECT:</b> Selects between the Unshielded Twisted Pair (UTP) transmit outputs (TXU+/-) and the Shielded Twisted Pair (STP) transmit outputs (TXS+/-).</p> <p>1 = UTP selected<br/>0 = STP selected</p> <p>Only one output pair (TXU+/- or TXS+/-) may be selected at one time. The pair that is not selected will tri-state.</p>                                                                                                 |
| 2    | LSS      | 0, RW   | <p><b>LOW SQUELCH SELECT:</b> Selects between standard 10BASE-T receiver squelch threshold and a reduced squelch threshold that is useful for longer cable applications and/or STP operation.</p> <p>1 = Low Squelch Threshold selected<br/>0 = Normal 10BASE-T Squelch Threshold selected</p>                                                                                                                                                        |
| 1    | Reserved | 0, RO   | <b>RESERVED:</b> Write as 0, read as don't care.                                                                                                                                                                                                                                                                                                                                                                                                      |
| 0    | JABEN    | 1, RW   | <p><b>JABBER ENABLE:</b> Enables or disables the Jabber function when the DP83840 is in 10BASE-T Full Duplex or 10BASE-T Transceiver Loopback mode (10BT_LPBK bit 11 in the LBREMR, address 18h).</p> <p>1 = Jabber function enabled<br/>0 = Jabber function disabled</p>                                                                                                                                                                             |

# 5.0 DP83840 Application

## 5.1 TYPICAL BOARD LEVEL APPLICATION

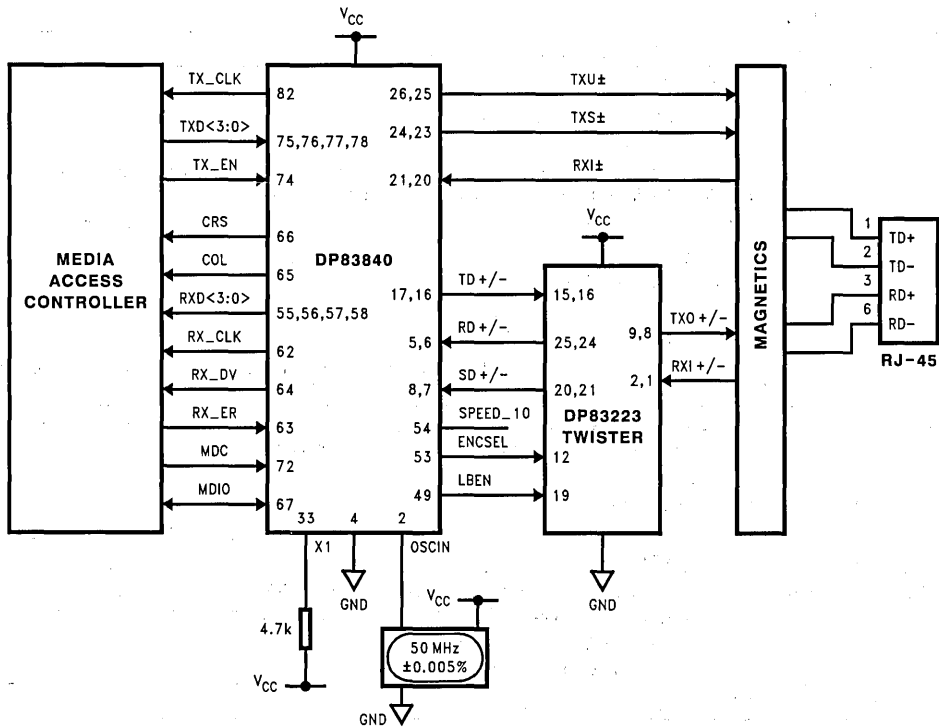


FIGURE 15. Typical Implementation of a 10/100 Mb/s Ethernet Node

TL/F/12388-15

Figure 15 shows a typical implementation of a 10/100 Mb/s Ethernet node application. This is given only to indicate the major circuit elements of such a design. It is not intended to be a full circuit diagram.

### 5.0 DP83840 Application (Continued)

The TD, SD and RD ECL differential signals must be terminated by a standard ECL load of 50Ω to a voltage source of 2V lower than V<sub>CC</sub> or the equivalent circuit: a Thevenin equivalent of 130Ω to GND accompanied by 82Ω to V<sub>CC</sub>.

### 5.2 LAYOUT RECOMMENDATIONS

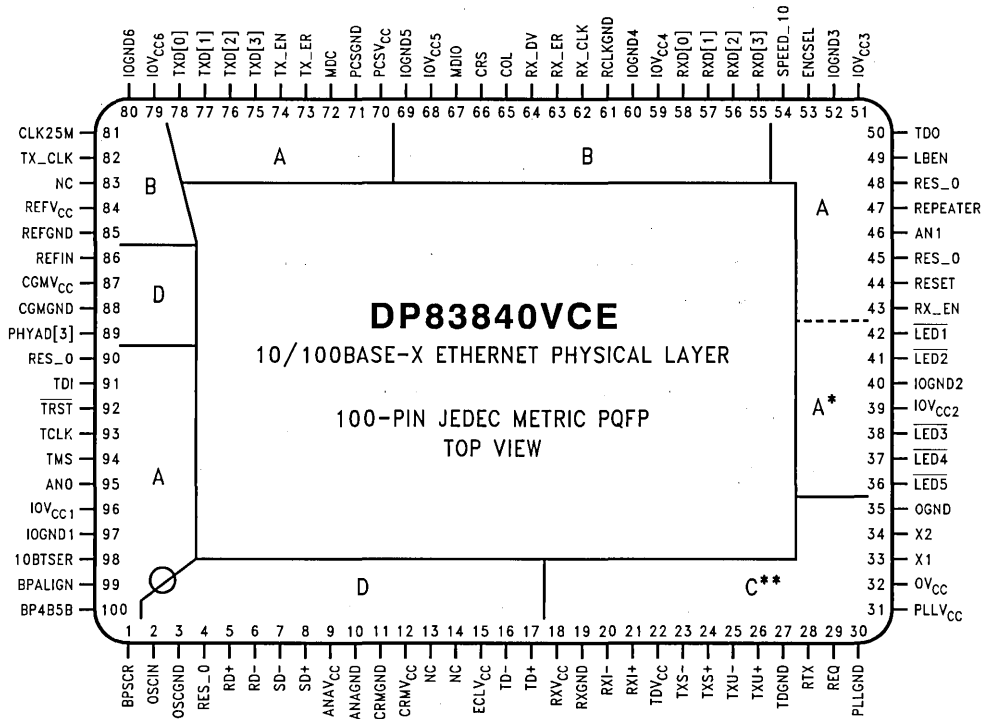
The V<sub>CC</sub> and Ground pins of the DP83840 are divided into 4 separate groups (as previously described in Section 2.10) to minimize ground bounce and cross talk as given in Table IV below.

TABLE IV. Supply Groupings

| Group | Description                                              |
|-------|----------------------------------------------------------|
| A     | TTL/CMOS Inputs V <sub>CC</sub> and GND Supply Pairs     |
| B     | TTL/CMOS Outputs V <sub>CC</sub> and GND Supply Pairs    |
| C     | 10BASE-T Circuitry V <sub>CC</sub> and GND Supply Pairs  |
| D     | 100BASE-X Circuitry V <sub>CC</sub> and GND Supply Pairs |

### 5.2.1 Signal Groupings vs V<sub>CC</sub> and GND Pairs

Figure 16 below shows how the signal pins are associated with the 4 groups of supply pins.



TL/F/12388-16

\*Pins 36 through 42 should preferably be grouped as "B" according to the table in Section 5.2, however, they are assigned as "A" to avoid further fragmentation of the power and ground planes. It is recommended that low current LEDs be used to minimize ground bounce caused by switching currents.

\*\*Group C requires that the PLL supply pins are further partitioned with a single point of return to the digital supply. See Section 5.3 for more details.

FIGURE 16. Power and Ground Grouping

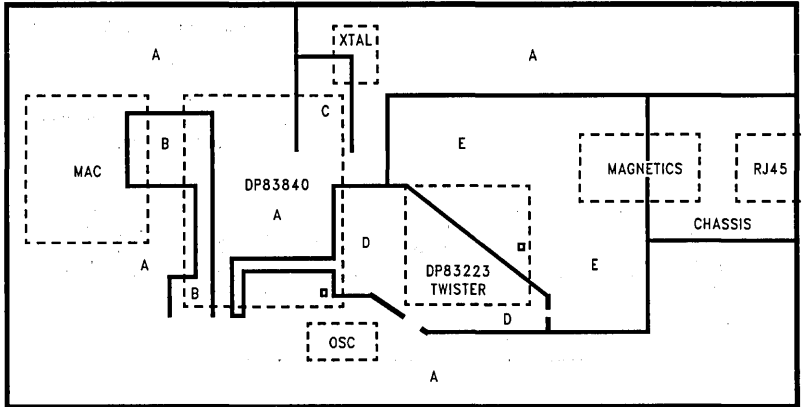
## 5.0 DP83840 Application (Continued)

### 5.3 PLANE PARTITIONING

The diagrams in *Figures 17 and 18* illustrate one approach to the partitioning of the power and ground planes at the board level. The A, B, C, and D plane notations reference the layout recommendations given in Section 5.2. The "E" plane notation shows additional partitioning that is required by the DP83223 Twister transceiver device.

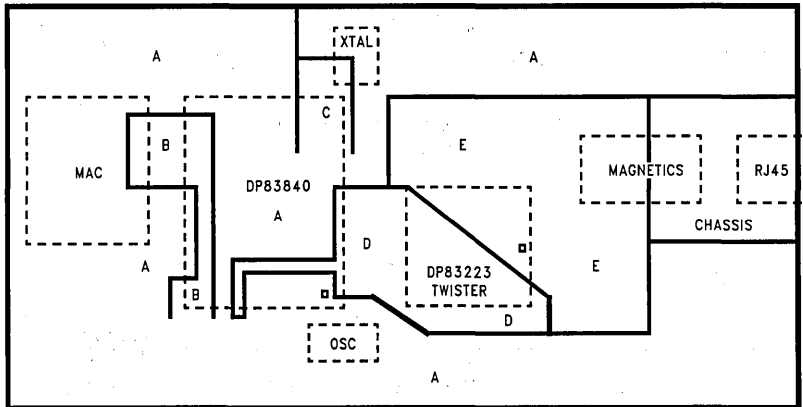
As indicated by the component placement in *Figure 17*, the distance between the 50 MHz oscillator (OSC) and the DP83840 should be minimized. This also applies to the 20 MHz crystal (XTAL) if it is included in the design.

Minimizing all signal trace lengths and using micro strip impedance control methods for the 100 Mb/s interconnections is recommended.



TL/F/12388-17

FIGURE 17. Recommended Ground Plane (GND) Partitioning



TL/F/12388-36

FIGURE 18. Recommended Power Plane (V<sub>CC</sub>) Partitioning



## 5.0 DP83840 Application (Continued)

### 5.4 POWER AND GROUND FILTERING

Sufficient filtering between the DP83840 power and ground pins placed as near to these pins as possible is recom-

mended. *Figure 19* suggests filtering for each power and ground pair as well as special consideration for the sensitive analog and PLL power pins.

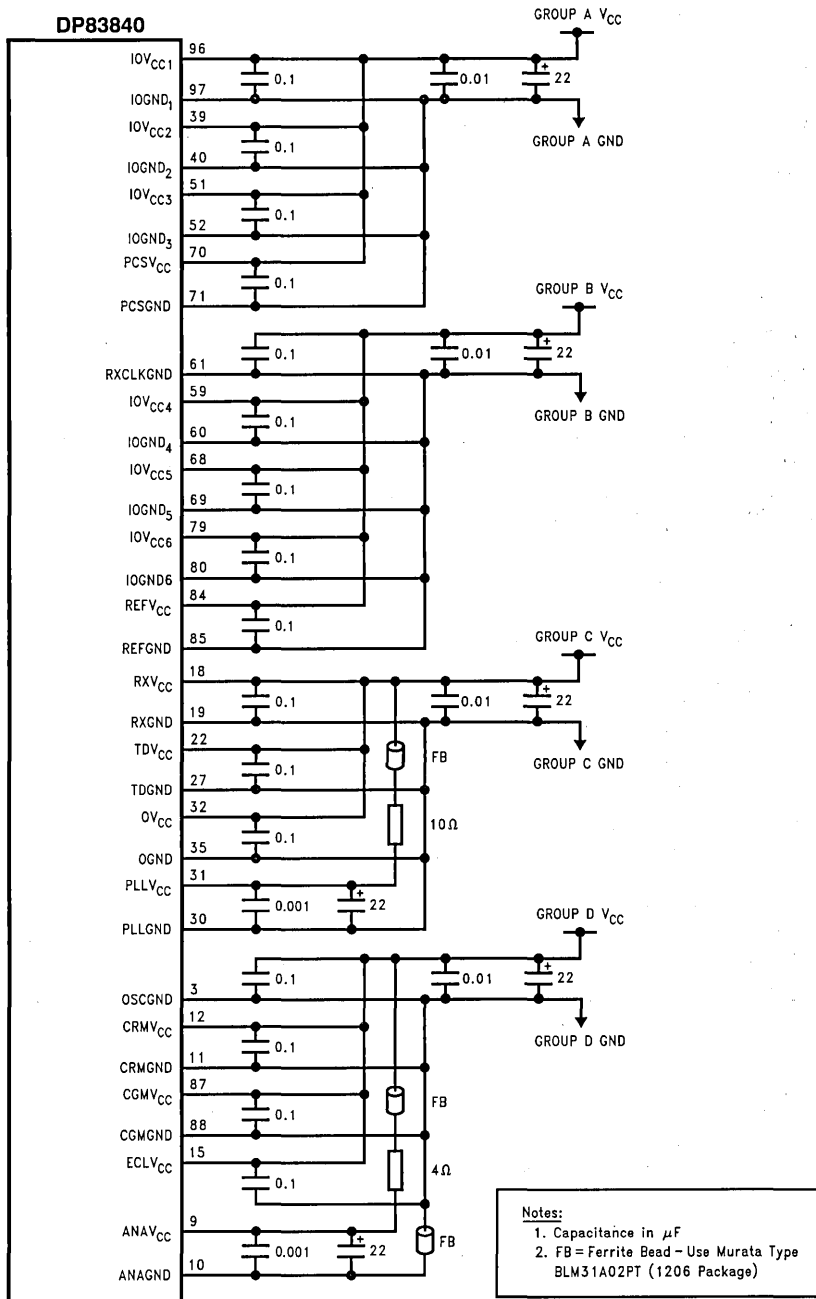


FIGURE 19. Recommended Power and Ground Filtering for Node Applications

TL/F/12388-18

## 6.0 DC and AC Specifications

### 6.1 RATINGS AND OPERATING CONDITIONS

#### 6.1.1 Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

|                               |                          |
|-------------------------------|--------------------------|
| Supply Voltage ( $V_{CC}$ )   | -0.5V to +7.0V           |
| Input Voltage ( $DC_{IN}$ )   | -0.5V to $V_{CC} + 0.5V$ |
| Output Voltage ( $DC_{OUT}$ ) | -0.5V to $V_{CC} + 0.5V$ |
| Storage Temperature           | -65°C to +150°C          |
| ECL Signal Output Current     | -50 mA                   |
| ESD Protection                | 2000V                    |

#### 6.1.2 Recommended Operating Conditions

|                                                              | Min  | Typ | Max  | Units |
|--------------------------------------------------------------|------|-----|------|-------|
| Supply Voltage ( $V_{CC}$ )                                  | 4.75 | 5.0 | 5.25 | V     |
| Ambient Temperature ( $T_A$ )                                | 0    |     | 70   | °C    |
| REFIN Input Frequency (25 MHz)                               | -50  |     | +50  | ppm   |
| REFIN Input Duty Cycle                                       | 35   |     | 65   | %     |
| OSCIN Input Frequency (50 MHz)                               | -50  |     | +50  | ppm   |
| OSCIN Input Duty Cycle                                       | 35   |     | 65   | %     |
| <b>Crystal Specifications:</b>                               |      |     |      |       |
| Crystal Center Frequency ( $X_{FC}$ )                        |      | 20  |      | MHz   |
| Crystal Freq. Calibration ( $X_{CAL}$ )                      | -10  |     | 10   | ppm   |
| Crystal Freq. Stability ( $X_{STAB}$ )<br>(Over Temperature) | -10  |     | 10   | ppm   |
| Crystal Aging ( $X_{AGING}$ )                                | -10  |     | 10   | ppm   |

### 6.2 DC SPECIFICATIONS

| Symbol   | Pin Types                             | Parameter               | Conditions               | Min                     | Typ | Max                     | Units         |
|----------|---------------------------------------|-------------------------|--------------------------|-------------------------|-----|-------------------------|---------------|
| $V_{IH}$ | I (except RX1 + / -)<br>I/O<br>I/O, Z | Input High Voltage      |                          | 2.0                     |     |                         | V             |
|          | ANO and AN1<br>Input Pins             |                         | $I_{IH} = 2 \text{ mA}$  | $V_{CC} - 1.0$          |     |                         | V             |
| $V_{IL}$ | I (except RX1 + / -)<br>I/O<br>I/O, Z | Input Low Voltage       |                          |                         |     | 0.8                     | V             |
|          | ANO and AN1<br>Input Pins             |                         | $I_{IL} = -2 \text{ mA}$ |                         |     | 1.0                     | V             |
| $V_{IM}$ | ANO and AN1<br>Inputs Only            | Input Mid Level Voltage | Pin<br>Unconnected       | $(V_{CC} \div 2) - 0.7$ | 2.5 | $(V_{CC} \div 2) + 0.7$ | V             |
| $I_{IH}$ | I<br>I/O<br>I/O, Z                    | Input High Current      | $V_{IN} = V_{CC}$        |                         |     | -10                     | $\mu\text{A}$ |
|          | X1 Input                              |                         | X2 = N.C.                |                         |     | -100                    |               |
| $I_{IL}$ | I<br>I/O<br>I/O, Z                    | Input Low Current       | $V_{IN} = \text{GND}$    |                         |     | 10                      | $\mu\text{A}$ |
|          | X1 Input                              |                         | X2 = N.C.                |                         |     | 100                     |               |
|          | TMS, TDI,<br>TRST Inputs              |                         |                          |                         |     | 1                       | mA            |
| $V_{OH}$ | O<br>O, Z<br>I/O<br>I/O, Z            | Output High Voltage     | $I_{OH} = -4 \text{ mA}$ | $V_{CC} - 0.5$          |     |                         | V             |
|          | TX_CLK Pin                            |                         |                          | $V_{CC} - 1.5$          |     |                         |               |
| $V_{OL}$ | O<br>O, Z<br>I/O<br>I/O, Z            | Output Low Voltage      | $I_{OL} = 4 \text{ mA}$  |                         |     | 0.4                     | V             |

## 6.0 DC and AC Specifications (Continued)

### 6.2 DC SPECIFICATIONS (Continued)

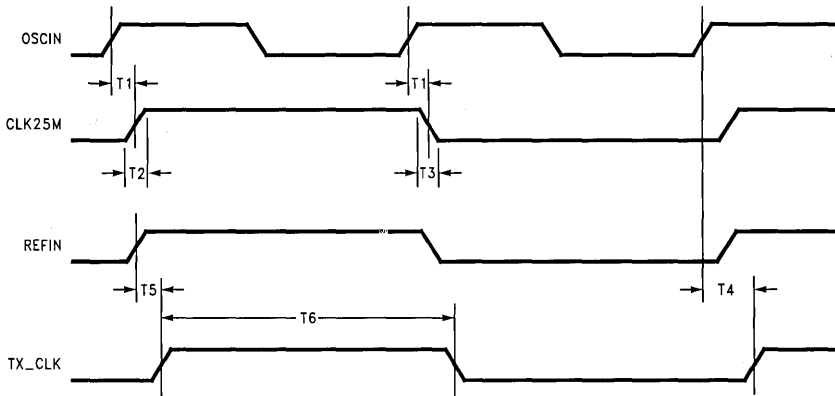
| Symbol             | Pin Types              | Parameter                   | Conditions                                              | Min                     | Typ | Max                     | Units |
|--------------------|------------------------|-----------------------------|---------------------------------------------------------|-------------------------|-----|-------------------------|-------|
| I <sub>OZ1</sub>   | I/O, Z<br>O, Z         | TRI-STATE Leakage           | V <sub>OUT</sub> = V <sub>CC</sub>                      |                         |     | 10                      | μA    |
| I <sub>OZ2</sub>   | I/O, Z<br>O, Z         | TRI-STATE Leakage           | V <sub>OUT</sub> = GND                                  |                         |     | -10                     | μA    |
| R <sub>OL</sub>    | TXU + / -<br>TXS + / - | Low Level Output Impedance  |                                                         |                         | 5   |                         | Ω     |
| R <sub>OH</sub>    | TXU + / -<br>TXS + / - | High Level Output Impedance |                                                         |                         | 5   |                         | Ω     |
| C <sub>IN</sub>    | I                      | Input Capacitance           |                                                         |                         |     |                         | pF    |
| C <sub>OUT</sub>   | O<br>Z                 | Output Capacitance          |                                                         |                         |     |                         | pF    |
| V <sub>TH1</sub>   | RXI + / -              | 10BASE-T Receive Threshold  |                                                         | ±300                    |     | ±585                    | mV    |
| V <sub>DIFF</sub>  | I (ECL)                | Input Voltage Differential  | Both Inputs Tested Together                             | 150                     |     |                         | mV    |
| V <sub>CM</sub>    | I (ECL)                | Common Mode Voltage         | Both Inputs Tested Together, V <sub>DIFF</sub> = 300 mV | V <sub>CC</sub> - 2.0   |     | V <sub>CC</sub> - 0.5   | V     |
| I <sub>INECL</sub> | I (ECL)                | Input Current               | V <sub>IN</sub> = V <sub>CC</sub> or GND                | -200                    |     | 200                     | μA    |
| V <sub>OHECL</sub> | O (ECL)                | Output High Voltage         | V <sub>IN</sub> = V <sub>IH max</sub>                   | V <sub>CC</sub> - 1.075 |     | V <sub>CC</sub> - 0.830 | V     |
| V <sub>OLECL</sub> | O (ECL)                | Output Low Voltage          | V <sub>IN</sub> = V <sub>IL max</sub>                   | V <sub>CC</sub> - 1.860 |     | V <sub>CC</sub> - 1.570 | V     |
| I <sub>CC</sub>    |                        | Total Supply Current        |                                                         |                         | 360 | 400                     | mA    |

## 6.0 DC and AC Specifications (Continued)

### 6.3 AC SPECIFICATIONS

#### 6.3.1 CGM Clock Timing

| Parameter | Description           | Notes                                                                             | Min  | Typ | Max  | Units |
|-----------|-----------------------|-----------------------------------------------------------------------------------|------|-----|------|-------|
| T1        | OSCIN to CLK25M Delay | OSCIN = 50 MHz                                                                    |      | 10  |      | ns    |
| T2        | CLK25M Rise Time      | 10% to 90%                                                                        |      | 5   |      | ns    |
| T3        | CLK25M Fall Time      | 90% to 10%                                                                        |      | 5   |      | ns    |
| T4        | OSCIN to TX_CLK Delay | 10 Mb/s Operation<br>(MII Nibble Mode)                                            |      | 10  |      | ns    |
| T4a       | OSCIN to TX_CLK Delay | 10 Mb/s Operation<br>(MII Serial Mode)                                            |      | 10  |      | ns    |
| T5        | REFIN to TX_CLK Delay | 100 Mb/s Operation                                                                | -3.0 |     | +3.0 | ns    |
| T6        | TX_CLK Duty Cycle     | 10 Mb/s Nibble (2.5 MHz),<br>10 Mb/s Serial (10 MHz),<br>100 Mb/s Nibble (25 MHz) | 35   |     | 65   | %     |

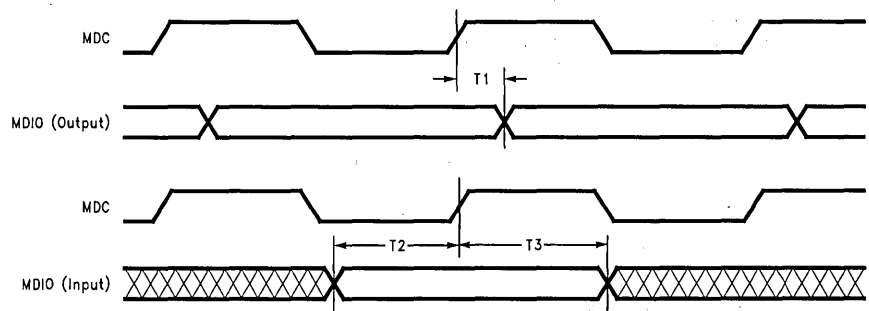


TL/F/12388-19

## 6.0 DC and AC Specifications (Continued)

### 6.3.2 MII Serial Management Timing

| Parameter | Description                     | Notes | Min | Typ | Max | Units |
|-----------|---------------------------------|-------|-----|-----|-----|-------|
| T1        | MDC to MDIO (Output) Delay Time |       | 0   |     | 300 | ns    |
| T2        | MDIO (Input) to MDC Set Time    |       | 10  |     |     | ns    |
| T3        | MDIO (Input) to MDC Hold Time   |       | 10  |     |     | ns    |

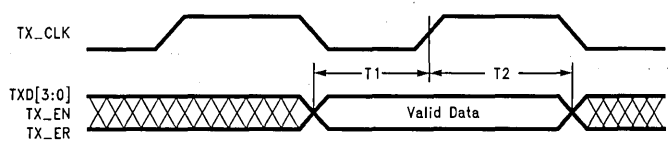


TL/F/12388-20

### 6.3.3 Transmit MII Timing

| Parameter | Description                             | Notes                        | Min | Typ | Max | Units |
|-----------|-----------------------------------------|------------------------------|-----|-----|-----|-------|
| T1        | TXD[3:0], TX_EN Data Setup to TX_CLK    | 10 Mb/s Nibble Mode          | 5   |     |     | ns    |
| T1a       | TXD[0], Data Setup to TX_CLK            | 10 Mb/s Serial Mode (Note 1) | 10  |     |     | ns    |
| T1b       | TX_EN Data Setup to TX_CLK              | 10 Mb/s Serial Mode (Note 1) | 30  |     |     | ns    |
| T1c       | TXD, TX_EN, TX_ER Data Setup to TX_CLK  | 100 Mb/s Nibble Mode         | 5   |     |     | ns    |
| T2        | TXD, TX_EN Data Hold from TX_CLK        | 10 Mb/s Nibble Mode          | 3   |     |     | ns    |
| T2a       | TXD[0] Data Hold from TX_CLK            | 10 Mb/s Serial Mode (Note 1) | 5   |     |     | ns    |
| T2b       | TX_EN Data Hold from TX_CLK             | 10 Mb/s Serial Mode (Note 1) | 5   |     |     | ns    |
| T2c       | TXD, TX_EN, TX_ER Data Hold from TX_CLK | 100 Mb/s Nibble Mode         | 3   |     |     | ns    |

**Note 1:** The 10 Mb/s serial mode of operation is an additional feature of the DP83840 and is independent of the MII specifications.



TL/F/12388-21

## 6.0 DC and AC Specifications (Continued)

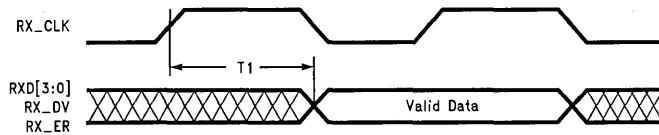
### 6.3.4 Receive MII Timing

| Parameter | Description                            | Notes                                                        | Min | Typ | Max | Units |
|-----------|----------------------------------------|--------------------------------------------------------------|-----|-----|-----|-------|
| T1        | RX_CLK to RXD, RX_DV Delay             | 10 Mb/s Nibble, 10 Mb/s Serial, 100 Mb/s Nibble (Notes 1, 2) | 25  |     | 35  | ns    |
| T1a       | RX_CLK to RXD[3:0], RX_DV, RX_ER Delay | 100 Mb/s BP_ALIGN Mode (Note 3)                              | -5  |     | 6   | ns    |

**Note 1:** RXD[3:0], RX\_DV, and RX\_ER are clocked out of the DP83840 on the falling edge of RX\_CLK. However, in order to specify this parameter without the RX\_CLK duty cycle affecting it, the timing is taken from the previous rising edge of RX\_CLK.

**Note 2:** The 10 Mb/s serial mode of operation is an additional feature of the DP83840 and is independent of the MII specifications.

**Note 3:** While in the BP\_ALIGN mode of operation, RXD[3:0], RX\_DV and RX\_ER are clocked out of the DP83840 on the rising edge of RX\_CLK. Refer to Section 3.1.1.4.



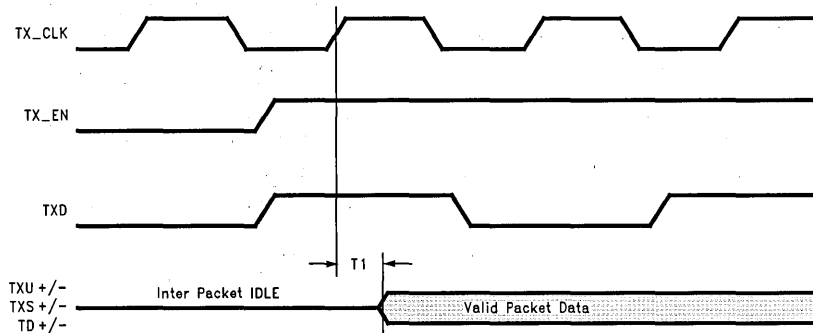
TL/F/12388-22

### 6.3.5 Transmit Packet Timing (Start of Packet)

| Parameter | Description              | Notes        | Min | Typ | Max | Units |
|-----------|--------------------------|--------------|-----|-----|-----|-------|
| T1        | TX_CLK to TXU+/- Latency |              |     |     | 10  | bits  |
| T1a       | TX_CLK to TD+/- Latency  | (Notes 1, 2) |     |     | 10  | bits  |

**Note 1:** 100BASE-TX Transmit Latency maximum limit equals 14 bit times.

**Note 2:** The latency of the DP83840 combined with the DP83223/A equals 11 bit times.

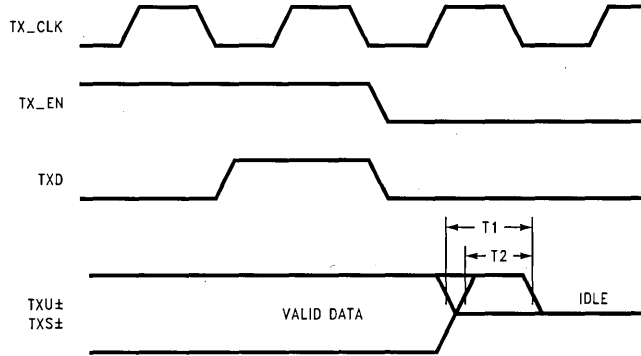


TL/F/12388-23

## 6.0 DC and AC Specifications (Continued)

### 6.3.6 Transmit Packet Timing (End of Packet)

| Parameter | Description             | Notes                                             | Min | Typ | Max | Units |
|-----------|-------------------------|---------------------------------------------------|-----|-----|-----|-------|
| T1        | End of Packet High Time | "1" As the Last Bit of the Packet, (10 Mb/s Only) | 250 |     |     | ns    |
| T2        | End of Packet High Time | "0" As the Last Bit of the Packet, (10 Mb/s Only) | 250 |     |     | ns    |



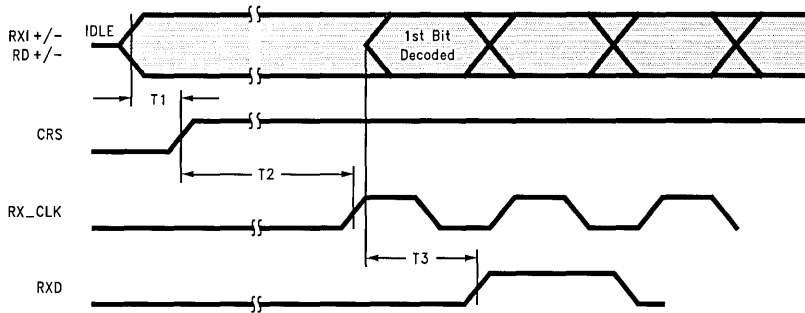
TL/F/12388-24

### 6.3.7 Receive Packet Timing (Start of Packet)

| Parameter | Description              | Notes                        | Min | Typ  | Max  | Units |
|-----------|--------------------------|------------------------------|-----|------|------|-------|
| T1        | Carrier Sense on Delay   | 10 Mb/s Nibble, Serial       |     |      | 10   | bits  |
| T1a       | Carrier Sense on Delay   | 100 Mb/s Nibble              |     |      | 10   | bits  |
| T2        | Decoder Acquisition Time | 10 Mb/s                      |     | 1100 | 2200 | ns    |
| T3a       | Receive Data Latency     | 10 Mb/s Nibble, Serial       |     |      | 20.3 | bits  |
| T3b       | Receive Data Latency     | 100 Mb/s Nibble (Notes 1, 2) |     |      | 20.3 | bits  |
| T3c       | Receive Data Latency     | 100 Mb/s BP_ALIGN            |     |      | 20.3 | bits  |

**Note 1:** 100BASE-TX Receive Latency maximum limits equals 23 bit times.

**Note 2:** The latency of the DP83840 combined with the DP83223 equals 21 bit times.

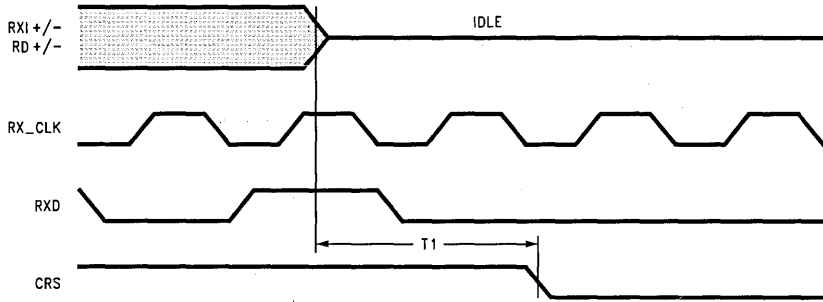


TL/F/12388-25

## 6.0 DC and AC Specifications (Continued)

### 6.3.8 Receive Packet Timing (End of Packet)

| Parameter | Description             | Notes    | Min | Typ | Max | Units |
|-----------|-------------------------|----------|-----|-----|-----|-------|
| T1        | Carrier Sense Off Delay | 10 Mb/s  |     |     | 8   | bits  |
| T1a       | Carrier Sense Off Delay | 100 Mb/s |     |     | 8   | bits  |

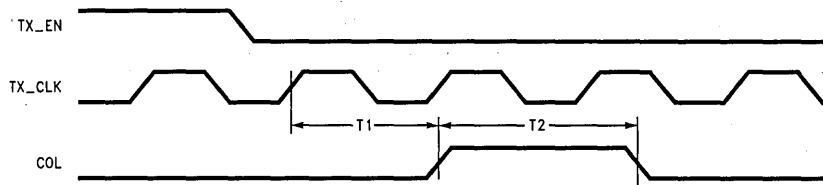


TL/F/12388-26

### 6.3.9 Heartbeat Timing

| Parameter | Description                         | Notes            | Min | Typ | Max | Units   |
|-----------|-------------------------------------|------------------|-----|-----|-----|---------|
| T1        | Collision Detect Heartbeat Delay    | 10 Mb/s (Note 1) | 0.6 |     | 1.6 | $\mu$ s |
| T2        | Collision Detect Heartbeat Duration | 10 Mb/s (Note 1) | 0.5 |     | 1.5 | $\mu$ s |

**Note 1:** The Heartbeat function operates during 10 Mb/s half-duplex. In 100 Mb/s half-duplex mode COL is used to indicate collisions. Since collisions cannot occur during full-duplex mode, the Heartbeat function is not implemented.



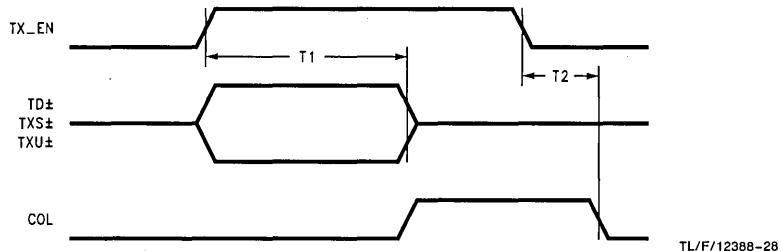
TL/F/12388-27



## 6.0 DC and AC Specifications (Continued)

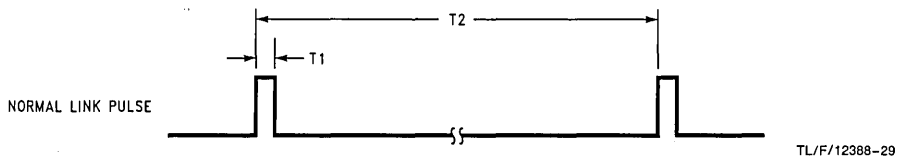
### 6.3.10 Jabber Timing

| Parameter | Description              | Notes               | Min | Typ | Max | Units |
|-----------|--------------------------|---------------------|-----|-----|-----|-------|
| T1        | Jabber Activation Time   | 10 Mb/s Half Duplex |     | 26  |     | ms    |
| T2        | Jabber Deactivation Time | 10 Mb/s Half Duplex |     | 410 |     | ms    |



### 6.3.11 10BASE-T Normal Link Pulse Timing

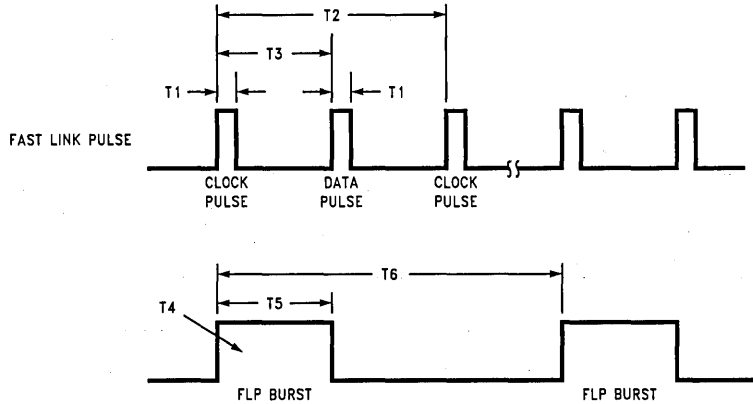
| Parameter | Description              | Notes | Min | Typ | Max | Units |
|-----------|--------------------------|-------|-----|-----|-----|-------|
| T1        | Normal Link Pulse Width  |       |     | 100 |     | ns    |
| T2        | Normal Link Pulse Period |       | 8   |     | 24  | ms    |



## 6.0 DC and AC Specifications (Continued)

### 6.3.12 Auto-Negotiation Fast Link Pulse (FLP) Timing

| Parameter | Description                       | Notes    | Min  | Typ | Max  | Units   |
|-----------|-----------------------------------|----------|------|-----|------|---------|
| T1        | Clock, Data Pulse Width           |          |      | 100 |      | ns      |
| T2        | Clock Pulse to Clock Pulse Period |          | 111  | 125 | 139  | $\mu$ s |
| T3        | Clock Pulse to Data Pulse Period  | Data = 1 | 55.5 |     | 69.5 | $\mu$ s |
| T4        | Number of Pulses in a Burst       |          | 17   |     | 33   | #       |
| T5        | Burst Width                       |          |      | 2   |      | ms      |
| T6        | FLP Burst to FLP Burst Period     |          | 8    |     | 24   | ms      |

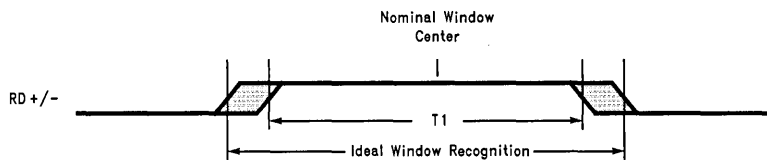


TL/F/12388-30

### 6.3.13 CRM Window Timing

| Parameter | Description                   | Notes    | Min | Typ | Max | Units |
|-----------|-------------------------------|----------|-----|-----|-----|-------|
| T1        | CRM Window Recognition Region | (Note 1) | -3  |     | 3   | ns    |

Note 1: The Ideal window recognition region is  $\pm 4$  ns.

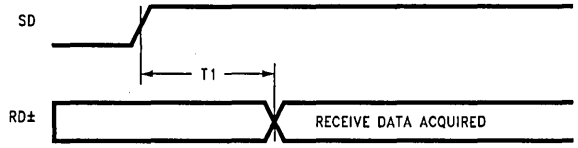


TL/F/12388-31

## 6.0 DC and AC Specifications (Continued)

### 6.3.14 CRM Acquisition Timing

| Parameter | Description                        | Notes            | Min | Typ | Max | Units         |
|-----------|------------------------------------|------------------|-----|-----|-----|---------------|
| T1        | CRM Receive Clock Acquisition Time | With SD Asserted |     |     | 100 | $\mu\text{s}$ |



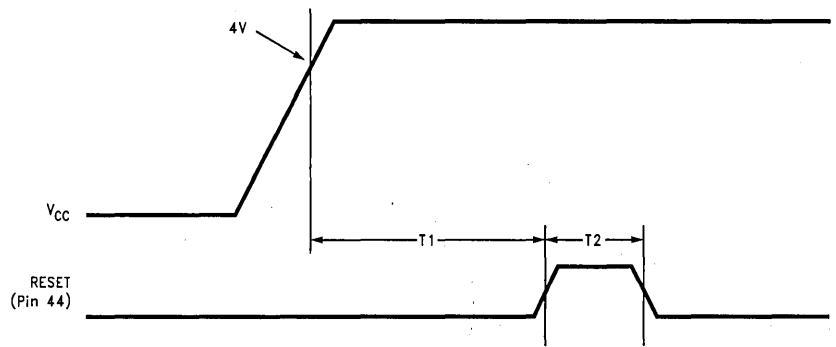
TL/F/12388-32

### 6.3.15 Reset Timing

| Parameter | Description                | Notes    | Min | Typ | Max | Units         |
|-----------|----------------------------|----------|-----|-----|-----|---------------|
| T1        | Internal Reset Time        | (Note 1) | 500 |     |     | $\mu\text{s}$ |
| T2        | Hardware RESET Pulse Width | (Note 2) | 1   |     |     | $\mu\text{s}$ |

**Note 1:** This timing assumes the use of a RESET pulse as opposed to RESET assertion immediately upon power-up. Either way, the deassertion of RESET must occur no sooner than 500  $\mu\text{s}$  after initial power-up.

**Note 2:** The Hardware Configuration pin values will be latched into the device no later than 2  $\mu\text{s}$  after the falling edge of RESET (refer to Section 3.10 for further detail).



TL/F/12388-37

# DP83850 100 Mb/s Repeater Interface Controller

## General Description

The DP83850 100 Mb/s Repeater Interface Controller is designed specifically to meet the needs of today's high speed Ethernet networking systems. The DP83850 is fully IEEE 802.3 clause 27 repeater compatible.

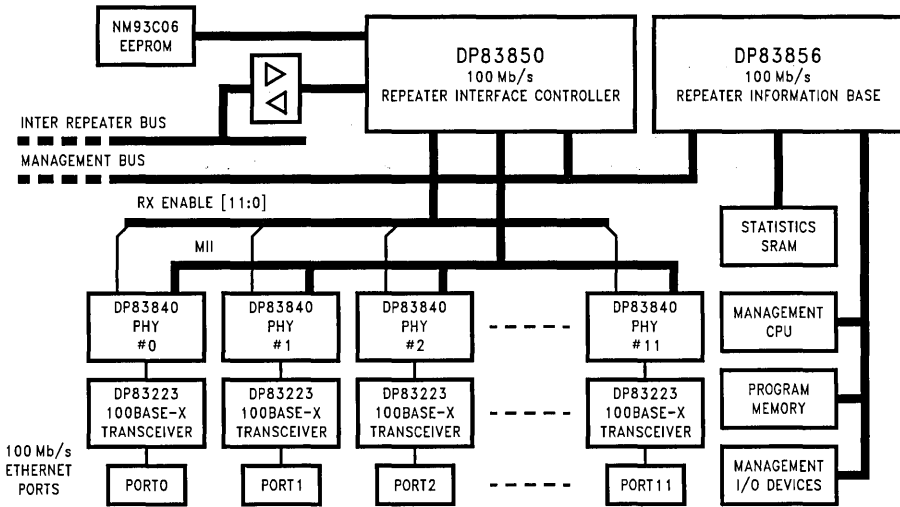
The DP83850 directly supports up to twelve 100 Mb/s links with its network interface ports. Larger repeaters with up to 372 ports may be constructed by cascading DP83850s together using the built in Inter Repeater bus.

In conjunction with a DP83856 100 Mb/s Repeater Information Base device, a DP83850 based repeater becomes a managed entity that is compatible with IEEE 802.3u, collecting and providing an easy interface to all the required network statistics.

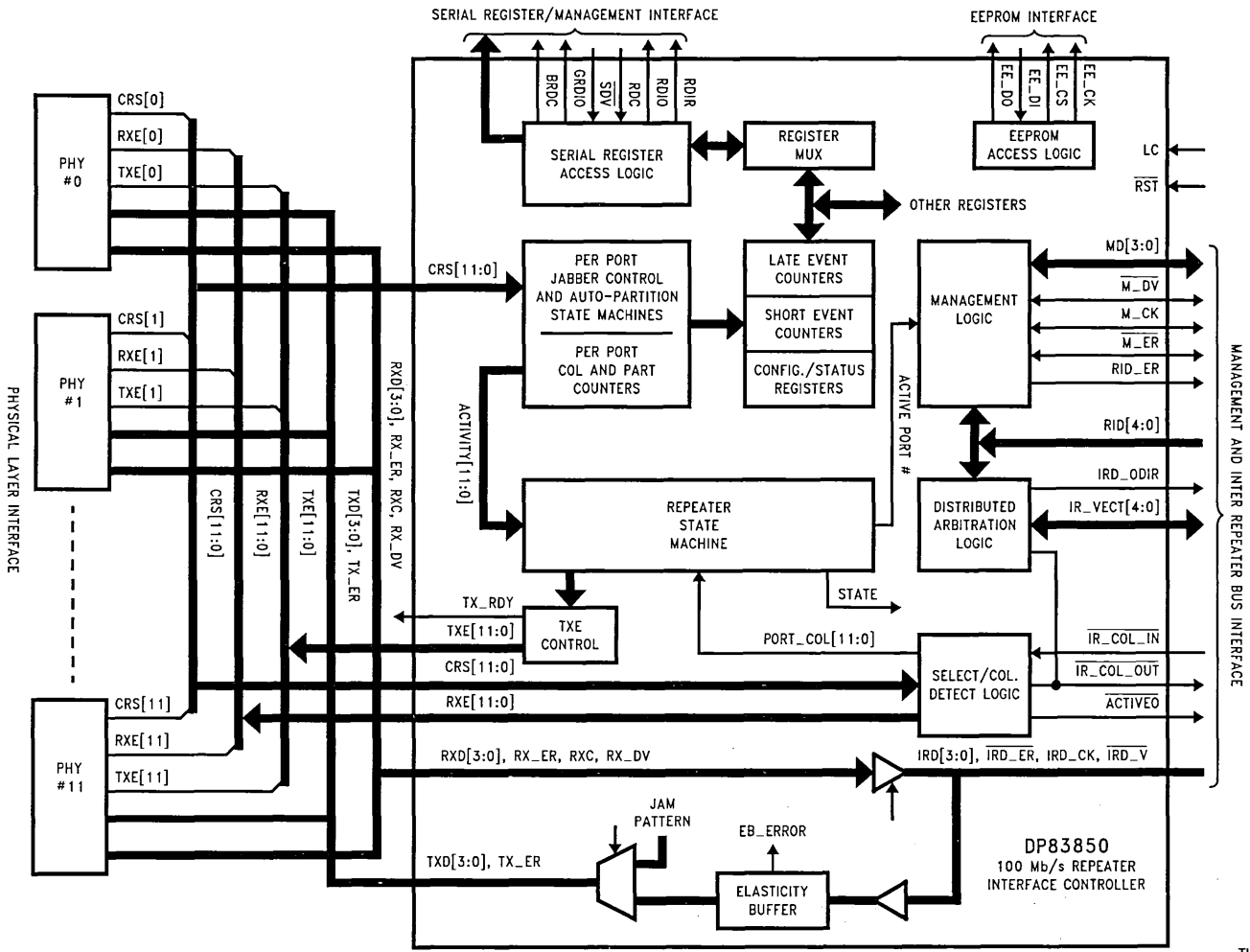
## Features

- IEEE 802.3u repeater and management compatible
- Supports 12 network connections (ports)
- Up to 31 repeater chips cascadable for larger hub applications (up to 372 ports)
- Separate jabber and partition state machines for each port
- Management interface to DP83856 allows all repeater MIBs to be maintained
- Large per-port management counters—reduces management CPU overhead
- On-chip elasticity buffer for PHY signal re-timing to the DP83850 clock source
- Serial register interface—reduces cost
- Physical layer device control/status access available via the serial register interface
- Detects repeater identification errors
- 132-pin PQFP

## System Diagram



TL/F/12391-1



TL/F/12391-2

## Table of Contents

### 1.0 PIN CONNECTION DIAGRAM

### 2.0 PIN DESCRIPTION

- 2.1 Physical Layer Interface
- 2.2 Inter Repeater and Management Bus Interface
- 2.3 EEPROM Interface
- 2.4 Miscellaneous
- 2.5 Pin Type Designation

### 3.0 FUNCTIONAL DESCRIPTION

- 3.1 Repeater State Machine
- 3.2 RXE Control
- 3.3 TXE Control
- 3.4 Data Path
- 3.5 Jabber Protection State Machine
- 3.6 Auto-Partition State Machine
- 3.7 Inter Repeater Bus Interface
- 3.8 Management Bus
- 3.9 Management Event Flags and Counters
- 3.10 Serial Register Interface
- 3.11 Jabber/Partition LED Driver Logic
- 3.12 EEPROM Serial Read Access

### 4.0 REGISTERS

- 4.1 Page 0 Register Map
- 4.2 Page 1 Register Map
- 4.3 Configuration Register (CONFIG)
- 4.4 Page Register (PAGE)
- 4.5 Partition Status Register (PARTITION)
- 4.6 Jabber Status Register (JABBER)
- 4.7 Administration Register (ADMIN)
- 4.8 Device ID Register (DEVICEID)
- 4.9 Hub ID 0 Register (HUBID0)
- 4.10 Hub ID 1 Register (HUBID1)
- 4.11 Port Management Counter Registers
- 4.12 Silicon Revision Register (SIREV)

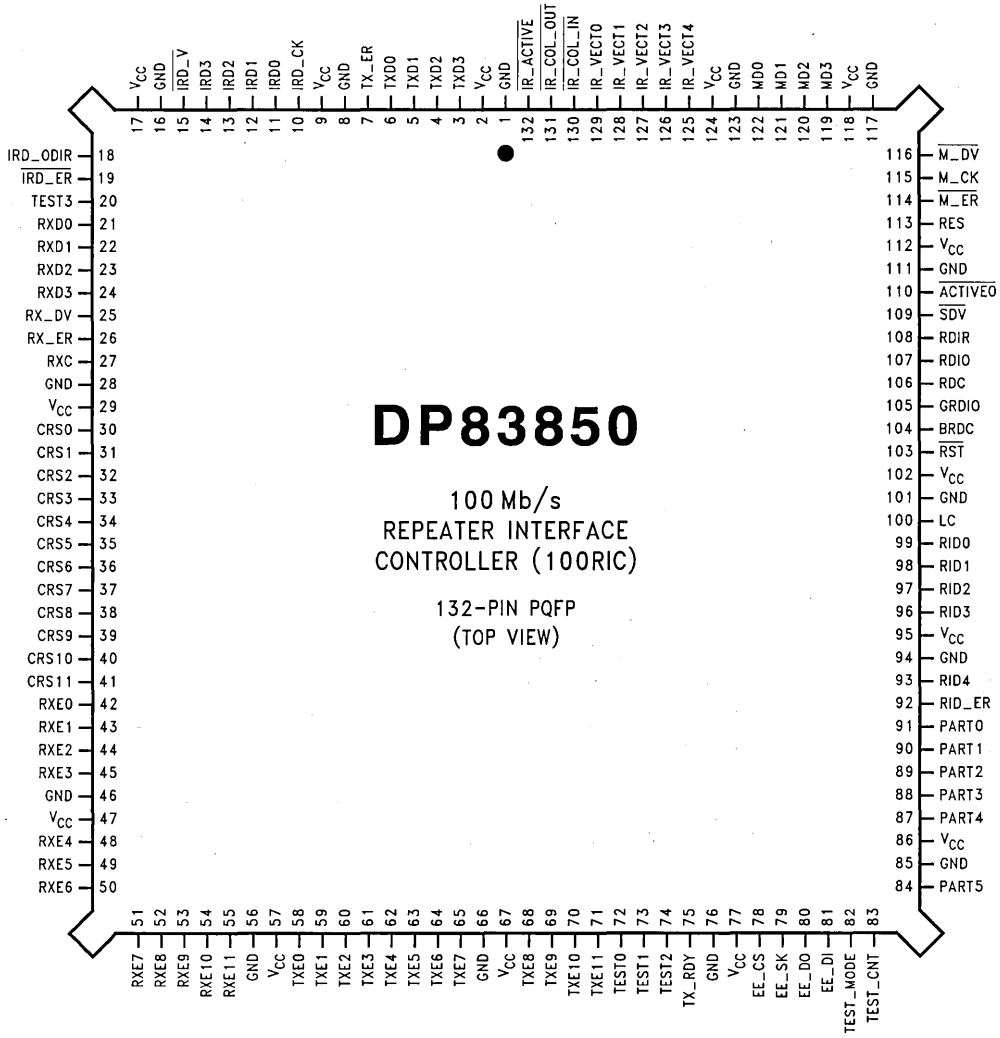
### 5.0 DP83850 APPLICATIONS

- 5.1 MII Interface Connections
- 5.2 Repeater ID Interface
- 5.3 Inter Repeater Bus Connections
- 5.4 DP83856 100RIB Connections
- 5.5 Port Partition and Jabber Status LEDs

### 6.0 AC AND DC SPECIFICATIONS

- 6.1 DC Specifications
- 6.2 AC Specifications

# 1.0 Pin Connection Diagram



TL/F/12391-3

## 2.0 Pin Description

### 2.1 PHYSICAL LAYER INTERFACE

| Signal Name | Type        | Active                          | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
|-------------|-------------|---------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------|----------|-------------------------|---|------|-----------------------|---|----|-----------------------|---|-------------|---------------------------------|---|----|---------------------------|---|-------------|----------|---|-------------|---------------------|
| RXD[3:0]    | I           |                                 | <b>RECEIVE DATA:</b> Nibble data inputs from each Physical layer chip. Up to 12 ports are supported.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
| RXE[11:0]   | O, L        | High                            | <b>RECEIVE ENABLE:</b> Asserted high to the respective Physical Layer chip to enable its Receive Data.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
| RX_DV       | I           | High                            | <b>RECEIVE DATA VALID:</b> Asserted High when valid data is present on RXD[3:0].                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
| RX_ER       | I           | High                            | <p><b>RECEIVE ERROR:</b> The physical Layer asserts this signal high when it detects receive error. When this signal is asserted, the DP83840 Physical Layer device indicates the type of error on RXD[3:0] as shown below. Note that this data is passed only to the Inter Repeater Bus, and not onto the TX Bus:</p> <table border="1"> <thead> <tr> <th>RX_ER</th> <th>RXD[3:0]</th> <th>Receive Error Condition</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>data</td> <td>Normal data reception</td> </tr> <tr> <td>1</td> <td>0h</td> <td>Symbol code violation</td> </tr> <tr> <td>1</td> <td>1h (Note 1)</td> <td>Elasticity Buffer Over/Underrun</td> </tr> <tr> <td>1</td> <td>2h</td> <td>Invalid Frame Termination</td> </tr> <tr> <td>1</td> <td>3h (Note 2)</td> <td>Reserved</td> </tr> <tr> <td>1</td> <td>4h (Note 2)</td> <td>10 Mb Link Detected</td> </tr> </tbody> </table> <p><b>Note 1:</b> The DP83840 must be configured with the Elasticity Buffer bypassed; hence this error code will never be generated.</p> <p><b>Note 2:</b> These error codes will only appear when CRS from the DP83840 is not asserted. Since the DP83850 only enables a DP83840 when its CRS is asserted, these error codes will never be passed through the chip.</p> | RX_ER | RXD[3:0] | Receive Error Condition | 0 | data | Normal data reception | 1 | 0h | Symbol code violation | 1 | 1h (Note 1) | Elasticity Buffer Over/Underrun | 1 | 2h | Invalid Frame Termination | 1 | 3h (Note 2) | Reserved | 1 | 4h (Note 2) | 10 Mb Link Detected |
| RX_ER       | RXD[3:0]    | Receive Error Condition         |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
| 0           | data        | Normal data reception           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
| 1           | 0h          | Symbol code violation           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
| 1           | 1h (Note 1) | Elasticity Buffer Over/Underrun |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
| 1           | 2h          | Invalid Frame Termination       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
| 1           | 3h (Note 2) | Reserved                        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
| 1           | 4h (Note 2) | 10 Mb Link Detected             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
| RXC         | I           |                                 | <b>RECEIVE CLOCK:</b> Recovered clock from the Physical Layer device. RXD, RX_DV, and RX_ER are generated from the falling edge of this clock.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
| CRS[11:0]   | I           | High                            | <b>CARRIER SENSE:</b> Asynchronous carrier indication from the Physical Layer device.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
| TXE[11:0]   | O, L        | High                            | <b>TRANSMIT ENABLE:</b> Enables corresponding port for transmitting data.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
| TX_RDY      | O, L        | High                            | <b>TRANSMIT READY:</b> Indicates when a transmit is in progress. Essentially, this signal is the logical "OR" of all TXEs.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
| TX_ER       | O, M        | High                            | <b>TRANSMIT ERROR:</b> Asserted high when a code violation is requested to be transmitted.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |
| TXD[3:0]    | O, H        | High                            | <b>TRANSMIT DATA:</b> Nibble data output to be transmitted by each Physical Layer device.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |       |          |                         |   |      |                       |   |    |                       |   |             |                                 |   |    |                           |   |             |          |   |             |                     |

**Note:** A table showing pin type designation is given in Section 2.5.



## 2.0 Pin Description (Continued)

### 2.2 INTER REPEATER AND MANAGEMENT BUS INTERFACE

| Signal Name                      | Type      | Active | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------------------------|-----------|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| IRD[3:0]                         | I/O/Z, L  |        | <b>INTER REPEATER DATA:</b> Nibble data input/output. Transfers data from the "active" DP83850 to all other "inactive" DP83850s. The busmaster of the IRD bus is determined by IR_VECT bus arbitration.                                                                                                                                                                                                                                                                                             |
| $\overline{\text{IRD\_ER}}$      | I/O/Z, L  | Low    | <b>INTER REPEATER DATA ERROR:</b> This signal carries the RX_ER state across the Inter Repeater bus. Used to track receive errors from the physical layer in real-time.                                                                                                                                                                                                                                                                                                                             |
| $\overline{\text{IRD\_V}}$       | I/O/Z, L  | Low    | <b>INTER REPEATER DATA VALID:</b> This signal carries the inverted RX_DV state across the Inter Repeater bus. It is used to frame good packets.                                                                                                                                                                                                                                                                                                                                                     |
| IRD_CK                           | I/O/Z, L  |        | <b>INTER REPEATER DATA CLOCK:</b> All Inter Repeater signals are synchronized to the rising edge of this clock.                                                                                                                                                                                                                                                                                                                                                                                     |
| IRD_ODIR                         | O, L      | High   | <b>INTER REPEATER DATA OUTWARD DIRECTION:</b> This pin indicates the direction of data for an external transceiver. It is HIGH when IRD[3:0], $\overline{\text{IRD\_V}}$ , IRD_CK, and $\overline{\text{IRD\_ER}}$ are driven out towards the Inter Repeater bus, and LOW when data is being received from the bus.                                                                                                                                                                                 |
| $\overline{\text{IR\_ACTIVE}}$   | I/O/OC, L | Low    | <b>INTER REPEATER ACTIVITY:</b> This "open-collector" type output is asserted when the repeater senses network activity.                                                                                                                                                                                                                                                                                                                                                                            |
| $\overline{\text{IR\_COL\_IN}}$  | I         | Low    | <b>INTER REPEATER COLLISION IN:</b> Indication from another DP83850 that it senses two or more ports receiving or another DP83850 has detected a collision.                                                                                                                                                                                                                                                                                                                                         |
| $\overline{\text{IR\_COL\_OUT}}$ | O/OC, L   | Low    | <b>INTER REPEATER COLLISION OUT:</b> Asserted when the DP83850 senses two or more ports receiving or non-idle, either within this DP83850 or in another DP83850, using the IR_VECT number to decide (if there is another device active, the IR_VECT number read will be different than the number this DP83850 currently has asserted).                                                                                                                                                             |
| IR_VECT[4:0]                     | I/O/OC, L | High   | <b>INTER REPEATER VECTOR:</b> When the repeater senses at least one of its ports active, it drives its unique vector (from RID[4:0]) onto these pins. If the vector value read back differs from its own (because another vector is being asserted by another device), this DP83850 will not drive the Inter Repeater signals and will de-assert its own vector value. If the value read back is the same as its own RID number this DP83850 will continue to drive the Inter Repeater bus signals. |
| MD[3:0]                          | I/O/Z, L  | High   | <b>MANAGEMENT DATA:</b> Outputs management information for the DP83856 management chip.                                                                                                                                                                                                                                                                                                                                                                                                             |
| $\overline{\text{M\_DV}}$        | I/O/Z, L  | Low    | <b>MANAGEMENT DATA VALID:</b> Asserted when valid data is present on MD[3:0].                                                                                                                                                                                                                                                                                                                                                                                                                       |
| M_CK                             | I/O/Z, L  |        | <b>MANAGEMENT CLOCK:</b> All data transfers on the management bus are synchronized to the rising edge of this clock.                                                                                                                                                                                                                                                                                                                                                                                |
| $\overline{\text{M\_ER}}$        | I/O/Z, L  | Low    | <b>MANAGEMENT ERROR:</b> Asserted when an Elasticity Buffer error has been detected.                                                                                                                                                                                                                                                                                                                                                                                                                |
| RDIO                             | I/O/Z, L  |        | <b>REGISTER DATA I/O:</b> Serial data input/output transfers data to/from the internal registers. Serial protocol conforms to the IEEE 802.3 $\mu$ MII (Media Independent Interface) specification.                                                                                                                                                                                                                                                                                                 |
| RDC                              | I         |        | <b>REGISTER DATA CLOCK:</b> All data transfers on RDIO are synchronized to the rising edge of this clock. RDC is limited to a maximum frequency of 2.5 MHz.                                                                                                                                                                                                                                                                                                                                         |
| $\overline{\text{SDV}}$          | I         | Low    | <b>SERIAL DATA VALID:</b> Asserted when a valid read or write command is present.                                                                                                                                                                                                                                                                                                                                                                                                                   |
| $\overline{\text{ACTIVEO}}$      | O/OC/L    | Low    | <b>ACTIVE OUT:</b> Enable for the IR_VECT[4:0] and $\overline{\text{IR\_ACTIVE}}$ signals. Used in multi-DP83850 systems to enable the external buffers driving these Inter Repeater Bus signals.                                                                                                                                                                                                                                                                                                   |

Note: A table showing pin type designation is given in Section 2.5.

## 2.0 Pin Description (Continued)

### 2.3 EEPROM INTERFACE

| Signal Name | Type | Active | Pin Description                                                        |
|-------------|------|--------|------------------------------------------------------------------------|
| EE_CS       | O, L | High   | EEPROM Chip Select: Asserted during reads to EEPROM                    |
| EE_CK       | O, L |        | EEPROM Serial Clock: Local Clock $\div$ 32 = 0.78125 MHz               |
| EE_DO       | I    |        | EEPROM Serial Data Out: Connected to the serial data out of the EEPROM |
| EE_DI       | O, L |        | EEPROM Serial Data In: Connected to the serial data in of the EEPROM   |

### 2.4 MISCELLANEOUS

| Signal Name | Type     | Active | Pin Description                                                                                                                                                                                                                                                                                                                                                                                               |
|-------------|----------|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LCK         | I        |        | <b>LOCAL CLOCK:</b> Must be run at 25 MHz. Used for TX data transfer to Physical Layer devices, TX Bus data transfers and DP83850 internal state machines.                                                                                                                                                                                                                                                    |
| RID[4:0]    | I        |        | <b>REPEATER IDENTIFICATION NUMBER:</b> Provides the unique vector for the IR_VECT[4:0] signals used in Inter Repeater bus arbitration. These bits are also used to uniquely identify this chip for serial register accesses.<br><br><b>Note:</b> The arbiter cannot use the value 1Fh as its arbitration vector. This is the IR_VECT[4:0] bus idle state, therefore RID[4:0] must never be set to this value. |
| RST         | I        | Low    | <b>RESET:</b> The chip is reset when this signal is asserted Low.                                                                                                                                                                                                                                                                                                                                             |
| GRDIO       | I/O/Z, L |        | <b>GATED REGISTER DATA INPUT/OUTPUT:</b> This I/O is a gated version of RDIO. When the "phy_access" bit in the CONFIG register is set High, the RDIO signal is passed through to GRDIO for accessing the physical layer chips.                                                                                                                                                                                |
| BRDC        | O, L     |        | <b>BUFFERED REGISTER DATA CLOCK:</b> Buffered version of RDC. Allows more devices to be chained on the MII serial bus.                                                                                                                                                                                                                                                                                        |
| RDIR        | O, L     | High   | <b>REGISTER DATA DIRECTION:</b> Direction signal for an external bi-directional buffer on the RDIO signal.<br><br>0 = RDIO data flows into the DP83850<br>1 = RDIO data flows out of the DP83850<br><br>Defaults to 0 when no register access is present.                                                                                                                                                     |
| PART[5:0]   | O, L     |        | <b>PARTITION:</b> Used to indicate each port's Jabber and Partition status. PART[3:0] cycle through each port number (0–12) continuously. PART[4] indicates the Partition status for each port (1 = Port Partitioned). PART[5] indicates the Jabber status for each port (1 = Port Jabbering). These pins are intended to be decoded to drive LEDs.                                                           |
| RID_ER      | O, L     | High   | <b>REPEATER ID ERROR:</b> This pin is asserted under the conditions which set the RID_error bit in the DEVICEID register.                                                                                                                                                                                                                                                                                     |
| TEST[3:0]   | O, L     |        | <b>TEST OUTPUTS:</b> No Connect.                                                                                                                                                                                                                                                                                                                                                                              |
| TEST_CNT    | I        | High   | <b>TEST CONTROL:</b> Factory test control input—tie Low for normal operation.                                                                                                                                                                                                                                                                                                                                 |
| TEST_MODE   | I        | High   | <b>TEST MODE:</b> Factory test mode input—tie Low for normal operation.                                                                                                                                                                                                                                                                                                                                       |
| RES         | O        |        | <b>RESERVED:</b> No Connect.                                                                                                                                                                                                                                                                                                                                                                                  |

**2.0 Pin Description** (Continued)**2.5 PIN TYPE DESIGNATION**

| Pin Type | Description                                                                                    |
|----------|------------------------------------------------------------------------------------------------|
| I        | Input Buffer                                                                                   |
| O        | Output Buffer, driven High or Low at all times                                                 |
| I/O/Z    | Bidirectional Buffer with High-impedance output                                                |
| O/Z      | Output Buffer with High-impedance capability                                                   |
| OC       | Open Collector like signals. These buffers are either driven low or in a high impedance state. |
| L        | Output Low drive: 4 mA                                                                         |
| M        | Output medium drive: 12 mA                                                                     |
| H        | Output High drive : 24 mA                                                                      |

## 3.0 Functional Description

The following sections describe the different functional blocks of the DP83850 100 Mb/s Repeater Interface Controller. The IEEE 802.3 repeater specification details a number of functions a repeater system must perform. These functions are split between those tasks that are common to all data channels and those that are specific to each individual channel. The DP83850 follows this split task approach for implementing the required functions.

### 3.1 REPEATER STATE MACHINE

The Repeater State Machine (RSM) is the main block that governs the overall operation of the repeater. At any one time, the RSM is in one of the following four states: Idle, Repeat, Collision, or One Port Left.

#### 3.1.1 Idle State

The RSM enters this state after reset or when there is no activity on the network and carrier sense is not present. The RSM exits this state if the above conditions are no longer true.

#### 3.1.2 Repeat State

This state is entered when there is a reception on only one of the ports, port N. While in this state, the data is transmitted to all the ports but the receiving port (port N). The RSM returns to Idle state when the reception ends, or transitions to Collision state if there is reception activity on more than one port.

#### 3.1.3 Collision State

When there is receive activity on more than one port of the repeater, the RSM moves to Collision state. In this state, transmit data is replaced by Jam and sent out to all ports including the original port N. There are two ways for the repeater to leave the Collision state. The first is when there is no receive activity on any of the ports. In this case, the repeater moves to Idle state. The second is when there is only one port experiencing collision in which case the repeater enters the One Port Left state.

#### 3.1.4 One Port Left State

This state is entered only from the Collision state. It guarantees that repeaters connected hierarchically will not jam each other indefinitely. While in this state, Jam is sent out to all ports except the port that has the receive activity. If more receive activity occurs on any other port, then the repeater moves to Collision state, otherwise it will transition to Idle state when the receive activity ends.

### 3.2 RXE CONTROL

When only one port has receive activity, the RXE signal (receive enable) is activated. If multiple ports are active, i.e. a collision scenario, then RXE will not be enabled for any port. The Port Select Logic asserts the open-collector outputs `IR_COLL_OUT` and `IR_ACTIVE` to indicate to other cascaded DP83850s that there is collision or receive activity present on this DP83850.

### 3.3 TXE CONTROL

This control logic enables the appropriate ports for data transmission according to the four states of the RSM. That is, during Idle state, no ports are enabled; during Repeat state, all ports but port N are enabled; in Collision state, all ports including port N are enabled; during One Port Left state all ports except the port experiencing the collision, will be enabled.

### 3.4 DATA PATH

After the Port Selection logic has enabled the active port, receive data (RXD), receive clock (RXC), receive error (`RX_ER`) and receive data valid (`RX_DV`) will flow through the chip from that port out onto the Inter Repeater (IR) bus if no collisions are present. The signals on the IR bus flow either in to or out of the chip depending upon the Repeater's state.

If the DP83850 is currently receiving and no collisions are present, the IR signals flow out of the chip. The DP83850's Arbitration Logic guarantees that only one DP83850 will gain ownership of the IR bus. In all other states, the IR signals are inputs.

When IR signals are inputs, the signals flow into the Elasticity Buffer (EB). Here, the data is re-timed and then sent out to the transmit ports. The Transmit Control logic determines which ports are enabled.

If a collision occurs a Jam pattern is sent out from the EB instead of the data. Jam pattern (3,4,3,4,..... from the DP83850, encoded by the Physical Layer device as 1,0,1,0,.....) is transmitted for the duration of the collision activity.

### 3.5 JABBER PROTECTION STATE MACHINE

The jabber specification for 100BASE-T is functionally different than 10BASE-T. In 100BASE-T, when a port jammers, its receive and transmit ports are cutoff until the jabber activity ceases. All other ports remain unaffected and continue normal operation. The jabber protect limit (i.e. the time for which a port can jabber until it is cutoff) for the DP83850 is 64k bit times.

### 3.6 AUTO-PARTITION STATE MACHINE

In order to protect the network from a port that is experiencing excessive consecutive collisions, each port has its own auto-partition state machine.

A port with excessive consecutive collisions will be partitioned after a programmed number of consecutive collisions occur on that port. Transmitting ports will not be affected.

The DP83850 has a configuration bit that allows the user to choose how many consecutive collisions a port should experience before partitioning. This bit can be set for either 32 or 64 consecutive collisions. A partitioned port will be reconnected when a collision-free packet of length 512 bits or more is received.

The DP83850 also provides a configuration bit that disables the auto-partition function completely.

### 3.7 INTER REPEATER BUS INTERFACE

The Inter Repeater bus is used to connect multiple DP83850s together to form a logical repeater unit and also to allow a managed entity. The IR bus allows received data packets to be transferred from the receiving DP83850 to the other DP83850s in the system. These DP83850s then send the data stream to their transmit enabled ports.

Notification of collisions to other cascaded DP83850s is as important as data transfer across the network. The IR bus has a set of status lines capable of conveying collision information between DP83850s to ensure their main state machines operate in the appropriate manner.

### 3.0 Functional Description (Continued)

The IR bus consists of the following signals:

- Inter Repeater Data. This is the transfer data, in nibble format, from the active DP83850 to all other cascaded DP83850s.
- Inter Repeater Data Error. This signal carries the receive error status from the physical layer in real-time.
- Inter Repeater Data Valid. This signal is used to frame good packets.
- Inter Repeater Data Clock. All IR data is synchronized to this clock.
- Inter Repeater Data Outward Direction. This pin indicates the direction of the data flow with respect to the DP83850. When the DP83850 is driving the IR bus (i.e. it contains port N) this signal is HIGH and when the DP83850 is receiving data from other DP83850s over the IR bus this signal is LOW.
- Inter Repeater Activity. When there is network activity the DP83850 asserts this output signal.
- Inter Repeater Collision Output. If there are multiple receptions on ports of a DP83850 or if the DP83850 senses concurrent activity on another DP83850 it asserts this output.
- Inter Repeater Collision Input. This input indicates that one of the cascaded DP83850s is experiencing a collision.
- Inter Repeater Vector. When there is reception on a port the DP83850 drives a unique vector onto these lines. The vector on the IR bus is compared with the Repeater ID (RID). The DP83850 will continue to drive the IR bus if both the vector and RID match.
- Active Output. This signal is asserted by a DP83850 when at least one of its ports is active. It is used to enable external bus transceivers.

#### 3.8 MANAGEMENT BUS

The task of network statistics gathering in a repeater system is divided between the DP83850 and DP83856 devices. Together, these devices collect all the required management information (compliant to IEEE 802.3u clause 30) associated with a packet.

Each time a packet is received by a DP83850, it drives the device and the port number onto the management bus in 3 contiguous nibbles of data. During a single reception, only one DP83850 drives this information onto the management bus. During a collision, the management bus will TRI-STATE as the information on this bus becomes invalid.

The first nibble of management data contains the least significant 4 bits of the RID number, the second contains the most significant bit of the RID number and the third contains the number of the receiving port.

The management bus also indicates whether an elasticity buffer error (due to under-run or over-run) has occurred by asserting the  $\overline{M\_ER}$  original.

#### 3.9 MANAGEMENT EVENT FLAGS AND COUNTERS

Repeater management statistics are supported either directly by using the DP83850's on-chip event flags and counters, or indirectly, by the DP83850 providing the information to the DP83856 via the management and transmit bus.

Management information is maintained within the DP83850 in two ways: event flags and counters.

##### 3.9.1 Event Flags

These are the events that provide a snapshot of the operation of the DP83850. These events include:

- Auto-Partition State, indicating whether a port is currently partitioned.
- Jabber State, indicating whether a port is in jabber state.
- Administration State, indicating if a port is disabled.

##### 3.9.2 Event Counters

The event counters maintain the statistics for events that occur too frequently for polled flags, or are collision oriented. Each port has its own set of event counters that keep track of the following events:

- Port Collisions. A 32-bit counter providing the number of collision occurrences on a port.
- Port Partitions. A 16-bit counter indicating the number of times that the port has partitioned.
- Late Events. A 32-bit counter indicating the number of times that a collision took place after 512-bit times (nominal). In the case of late events, both the late event and the collision counters will be incremented.
- Short Events. A 32-bit counter indicating the number of packets whose length is 76 bits (nominal) or less.

#### 3.10 SERIAL REGISTER INTERFACE

The DP83850 has 64 registers held in two pages of 32 (Register Page 0 and Register Page 1). The registers are 16 bits wide. Only one page of registers can be accessed at a time.

After power-up and/or reset, the DP83850 defaults to Register Page 0. Register Page 1 can be accessed by writing 0001h to the PAGE register in Register Page 0, whereupon further accesses will be to Register Page 1. Subsequently writing 0000h to the PAGE register in Register Page 1 switches the registers back to Register Page 0.

All accesses to DP83850 registers and counters, and to the connected Physical Layer devices (via the DP83850), are performed serially using the RDIO and RDC pins. The RDC clock is limited to a frequency no greater than 2.5 MHz. This interface implements the serial management protocol defined by the MII specification, IEEE 802.3u clause 22. The protocol uses bit streams with the following format:

For Read operation:

```
<start> <opcode> <device addr> <reg addr>
[turnaround] 0 <data>
```

For Write operation:

```
<start> <opcode> <device addr> <reg addr>
<10> <data>
```

This protocol allows for up to 32 devices (DP83850s or other MII compliant devices) to be connected, each with a unique address and up to 32 16-bit registers. Devices are cascaded on the RDIO and RDC signals.

### 3.0 Functional Description (Continued)

Since the RDIO pin is shared for both read and write operations, it must only be driven at the proper time. The serial protocol assumes that there is only one master (generally, the management entity's processor) and one or more slave devices (generally, the Physical Layer or DP83850 chips). The master drives RDIO at all times except when, during a slave read operation, the addressed slave places the serialized read data onto the RDIO line after the line turnaround field's first bit.

The fields of the protocol are defined in Table I. For the protocol to work, all serial logic must first be "synchronized" to incoming data. A preamble of 32 consecutive 1's transmitted before the <start> field ensures "data lock". This preamble only needs to be sent once (at reset). From then on, the <start> field lets the receive logic know where the beginning of the data frame occurs.

To access the Physical Layer devices via the serial bus, the DP83850 has a "phy\_access" mode. When in this mode, the register data input/output (RDIO) is gated to the GRDIO pin. This signal is connected to the serial data pins of the Physical Layer devices.

In this mode the buffers which drive RDIO and GRDIO will turn on in the appropriate direction for each serial access. In order to avoid possible contention problems the user must ensure that only one DP83850 at a time has the "phy\_access" bit set. The CONFIG register contains the "phy\_access" bit, which can be set or cleared at any time.

Figure 1 shows a possible system implementation of the RDIO/GRDIO connection scheme. In this example the DP83850 with address 00001 has its "phy\_access" bit set, allowing its twelve DP83840 PHY devices to be accessed by the DP83856 100RIB.

MII serial management contention problems can be avoided by keeping to the addressing convention shown in Figure 1.

#### 3.11 JABBER/PARTITION LED DRIVER LOGIC

This logic encodes the current auto-partition status (from the PARTITION register) and the jabber status (from the JABBER register), and outputs this information to PART[5:0] pins. PART[3:0] cycles through each port number and PART[5:4] indicates the port's status. PART[5] indicates the Jabber status for each port (0 = LED OFF, 1 = LED ON—Port Jabbering). PART[4] indicates the Partition status for each port (0 = LED OFF, 1 = LED ON—Port Auto-Partitioned).

The port number on PART[3:0] is cycled with a 25 MHz clock. External logic is required to decode the PART[5:0] outputs and drive the Partition and Jabber LEDs. Multi-color LEDs could be driven with the appropriate logic if required.

One possible implementation of a DP83850 Port Partition and Jabber Status LED scheme is given in Section 5.5.

#### 3.12 EEPROM SERIAL READ ACCESS

After reset is de-asserted, the DP83850 will serially read an NM93C06 EEPROM (or equivalent). Only the first 32 bits starting from address 0 will be read. Write access is not provided. The data is written to registers HUBID0 and HUBID1. The first bit read is written to HUBID0[0]; the last bit read will be written to HUBID1[15].

The DP83850 EEPROM interface implements the serial protocol as shown in Figure 2. The DP83850 will issue two read commands to obtain the 32-bit ID. The serial clock, EE\_\_CK will be continuous. For more explicit timing diagrams please refer to the NM93C06 datasheet.

The NM93C06 EEPROM must be pre-programmed with the HUBID value prior to fitting the device to the circuit board, since the DP83850 does not support programming of this device in circuit.

TABLE I. Serial Register Interface Encoding

| Field         | Encoding               | Description                                                  |
|---------------|------------------------|--------------------------------------------------------------|
| <start>       | 01                     | Indicates the beginning of an opcode operation.              |
| <opcode>      | 10<br>01<br>All Others | Read<br>Write<br>Reserved                                    |
| <reg addr>    | 00000–11111            | Five bits are provided to address up to 32 16-bit registers. |
| <device addr> | 00000–11111            | Five bits are provided to address up to 32 devices.          |

### 3.0 Functional Description

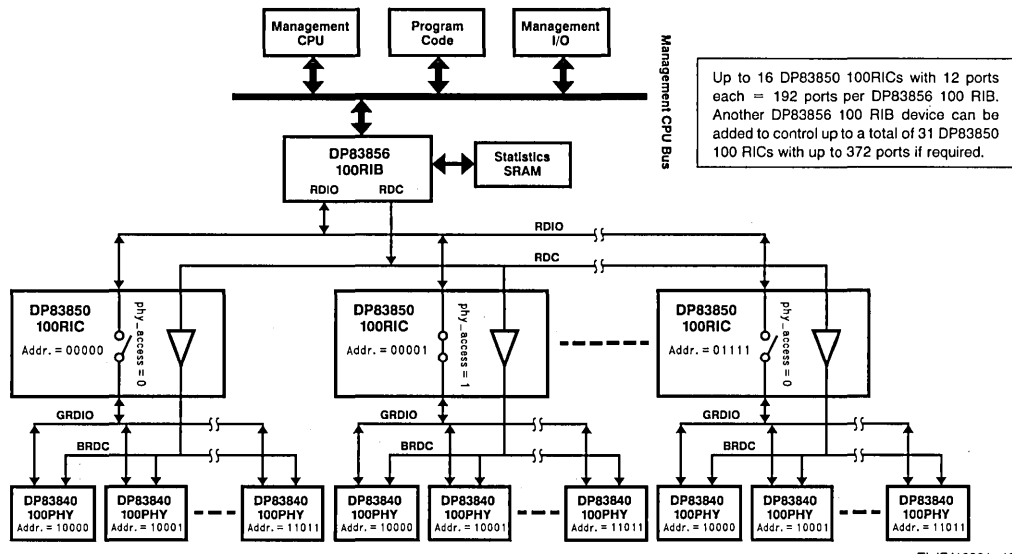


FIGURE 1. Serial Management Addressing Scheme

TL/F/12391-15

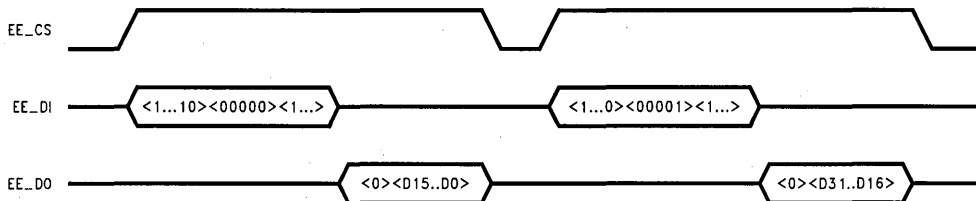


FIGURE 2. Serial EEPROM Access Protocol

TL/F/12391-4

## 4.0 Registers

The DP83850 has 64 registers in 2 pages of 32 16-bit registers. At power-on and/or reset, the DP83850 defaults to page 0 registers. The register page can be changed by writing to the PAGE register in either register page.

The register page maps are given in Sections 4.1 and 4.2, followed by a detailed description of the registers in Sections 4.3 to 4.12.

### 4.1 PAGE 0 REGISTER MAP

| Address (hex) | Name                   | Access    | Description                                                                                                                                                                                                                                                                         |
|---------------|------------------------|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 0             | CONFIG                 | R/W       | Sets the DP83850 configuration.                                                                                                                                                                                                                                                     |
| 1             | PAGE                   | R/W       | Selects either register page 0 or 1.                                                                                                                                                                                                                                                |
| 2             | PARTITION              | Read Only | Indicates Auto-Partitioning status.                                                                                                                                                                                                                                                 |
| 3             | JABBER                 | Read Only | Indicates Jabber status.                                                                                                                                                                                                                                                            |
| 4             | ADMIN                  | R/W       | Port enable/disable, administration control/status.                                                                                                                                                                                                                                 |
| 5             | DEVICEID               | R/W       | Accesses a) the DP83850 ID number configured externally on the RID[4:0] pins and b) the last receiving port number. The DP83850 device number may be overwritten after it has been latched at the end of reset: be careful not to have duplicate ID's on the same IR bus interface. |
| 6             | HUBID0                 | Read Only | First 16 bits read from EEPROM                                                                                                                                                                                                                                                      |
| 7             | HUBID1                 | Read Only | Second 16 bits read from EEPROM                                                                                                                                                                                                                                                     |
| 8             | P0__SE                 | R/W       | Port 0: 32-bit ShortEvent counter (See access rules in Section 4.11)                                                                                                                                                                                                                |
| 9             | P0__LE                 | R/W       | Port 0: 32-bit LateEvent counter (See access rules in Section 4.11)                                                                                                                                                                                                                 |
| A             | P0__COL                | R/W       | Port 0: 32-bit Collision counter (See access rules in Section 4.11)                                                                                                                                                                                                                 |
| B             | P0__PART               | R/W       | Port 0: 16-bit Auto-Partition counter                                                                                                                                                                                                                                               |
| C             | P1__SE                 | R/W       | Port 1: 32-bit ShortEvent counter (See access rules in Section 4.11)                                                                                                                                                                                                                |
| D             | P1__LE                 | R/W       | Port 1: 32-bit LateEvent counter (See access rules in Section 4.11)                                                                                                                                                                                                                 |
| E             | P1__COL                | R/W       | Port 1: 32-bit Collision counter (See access rules in Section 4.11)                                                                                                                                                                                                                 |
| F             | P1__PART               | R/W       | Port 1: 16-bit Auto-Partition counter                                                                                                                                                                                                                                               |
| 10-13         | P2__SE<br>... P2__PART | R/W       | Port 2 management counters (as per ports 0, 1 above)                                                                                                                                                                                                                                |
| 14-17         | P3__SE ...<br>P3__PART | R/W       | Port 3 management counters (as per ports 0, 1 above)                                                                                                                                                                                                                                |
| 18-1B         | P4__SE ...<br>P4__PART | R/W       | Port 4 management counters (as per ports 0, 1 above)                                                                                                                                                                                                                                |
| 1C-1F         | P5__SE<br>...P5__PART  | R/W       | Port 5 management counters (as per ports 0, 1 above)                                                                                                                                                                                                                                |



## 4.0 Registers (Continued)

### 4.2 PAGE 1 REGISTER MAP

| Address (hex) | Name                   | Access    | Description                                                      |
|---------------|------------------------|-----------|------------------------------------------------------------------|
| 0             | CONFIG                 | R/W       | Sets the DP83850 configuration (same as page 0 CONFIG register). |
| 1             | PAGE                   | R/W       | Select either register page 0 or 1.                              |
| 2             |                        |           | Reserved                                                         |
| 3             |                        |           | Reserved                                                         |
| 4             | SIREV                  | Read Only | Silicon revision code.                                           |
| 5-7           |                        |           | Reserved                                                         |
| 8-B           | P6_SE ...<br>P6_PART   | R/W       | Port 6 management counters (as per ports 0, 1 above)             |
| C-F           | P7_SE ...<br>P7_PART   | R/W       | Port 7 management counters (as per ports 0, 1 above)             |
| 10-13         | P8_SE ...<br>P8_PART   | R/W       | Port 8 management counters (as per ports 0, 1 above)             |
| 14-17         | P9_SE ...<br>P9_PART   | R/W       | Port 9 management counters (as per ports 0, 1 above)             |
| 18-1B         | P10_SE ...<br>P10_PART | R/W       | Port 10 management counters (as per ports 0, 1 above)            |
| 1C-1F         | P11_SE ...<br>P11_PART | R/W       | Port 11 management counters (as per ports 0,1 above)             |

## 4.0 Registers (Continued)

### 4.3 CONFIGURATION REGISTER (CONFIG)

Page 0 Address 0h

Page 1 Address 0h

| Bit    | Bit Name    | Access | Bit Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|--------|-------------|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D15–D5 | Reserved    |        | For compatibility with future enhanced versions these bits must be written as zero. They are undefined when read.                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| D4     | MGTEN       | R/W    | <p>Management Enable: This bit enables all the management counters.</p> <p>0: Management Counters disabled (default).<br/>1: Management Counters enabled.</p> <p><b>Note:</b> The management counters can only be reliably written to when they are disabled.</p>                                                                                                                                                                                                                                                                                                        |
| D3     | COL_LIMIT32 | R/W    | <p>This bit configures the collision limit for Auto-Partitioning:</p> <p>0: Consecutive Collision Limit set to 64 consecutive collisions (default). A port will be partitioned on the 65th consecutive collision.<br/>1: Consecutive Collision Limit set to 32 consecutive collisions. A port will be partitioned on the 33rd consecutive collision.</p>                                                                                                                                                                                                                 |
| D2     | DIS_PART    | R/W    | <p>This bit disables the Auto-Partitioning algorithm:</p> <p>0: Auto-Partitioning is not disabled (default).<br/>1: Auto-Partitioning is disabled.</p>                                                                                                                                                                                                                                                                                                                                                                                                                   |
| D1     | PHY_ACCESS  | R/W    | <p>This bit allows the management agent to access the DP83840 PHY chip's register via the MII serial protocol.</p> <p>0: PHY access disabled (default).<br/>1: PHY register access enabled.</p> <p><b>Note:</b> When in PHY access mode, RDIO will be driven by the DP83850 during the read phase for all read commands. This is to allow the DP83840 Physical Layer devices to pass their data through their local DP83850. While in this mode, contention will result (on the RDIO line) if any device other than the DP83840 Physical Layer devices are accessed.</p> |
| D0     | RST_RSM     | R/W    | <p>Setting this bit holds the Repeater State Machines in reset. The management event flags and counters are unaffected by this bit. Setting this bit while a reception is in progress may truncate the packet.</p> <p>0: DP83850 in normal operation (default).<br/>1: DP83850 held in reset.</p>                                                                                                                                                                                                                                                                        |

## 4.0 Registers (Continued)

### 4.4 PAGE REGISTER (PAGE)

Page 0 Address 1h

Page 1 Address 1h

| Bit       | Bit Name  | Access    | Bit Description                                                                                                                                                                                                                                                                                                                                                                                                         |           |      |  |    |   |           |    |   |  |    |          |  |    |          |  |
|-----------|-----------|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------|------|--|----|---|-----------|----|---|--|----|----------|--|----|----------|--|
| D15–D2    | Reserved  |           | These bits are undefined when read. Must be written as 0.                                                                                                                                                                                                                                                                                                                                                               |           |      |  |    |   |           |    |   |  |    |          |  |    |          |  |
| D1–D0     | PAGE[1:0] | R/W       | These bits program the register page to be accessed. The page encoding is as follows:<br><br><table border="1"> <thead> <tr> <th>PAGE[1:0]</th> <th>Page</th> <th></th> </tr> </thead> <tbody> <tr> <td>0h</td> <td>0</td> <td>(Default)</td> </tr> <tr> <td>1h</td> <td>1</td> <td></td> </tr> <tr> <td>2h</td> <td>Reserved</td> <td></td> </tr> <tr> <td>3h</td> <td>Reserved</td> <td></td> </tr> </tbody> </table> | PAGE[1:0] | Page |  | 0h | 0 | (Default) | 1h | 1 |  | 2h | Reserved |  | 3h | Reserved |  |
| PAGE[1:0] | Page      |           |                                                                                                                                                                                                                                                                                                                                                                                                                         |           |      |  |    |   |           |    |   |  |    |          |  |    |          |  |
| 0h        | 0         | (Default) |                                                                                                                                                                                                                                                                                                                                                                                                                         |           |      |  |    |   |           |    |   |  |    |          |  |    |          |  |
| 1h        | 1         |           |                                                                                                                                                                                                                                                                                                                                                                                                                         |           |      |  |    |   |           |    |   |  |    |          |  |    |          |  |
| 2h        | Reserved  |           |                                                                                                                                                                                                                                                                                                                                                                                                                         |           |      |  |    |   |           |    |   |  |    |          |  |    |          |  |
| 3h        | Reserved  |           |                                                                                                                                                                                                                                                                                                                                                                                                                         |           |      |  |    |   |           |    |   |  |    |          |  |    |          |  |

### 4.5 PARTITION STATUS REGISTER (PARTITION)

Page 0 Address 2h

| Bit     | Bit Name                | Access    | Bit Description                                                                                                                   |
|---------|-------------------------|-----------|-----------------------------------------------------------------------------------------------------------------------------------|
| D15–D12 | Reserved                |           | These bits are undefined when read.                                                                                               |
| D11–D0  | PART[11] ...<br>PART[0] | Read Only | The respective port's PART bit is set to 1 when Partitioning is sensed on that port. After reset, these bits are cleared to zero. |

### 4.6 JABBER STATUS REGISTER (JABBER)

Page 0 Address 3h

| Bit     | Bit Name   | Access    | Bit Description                                                                                                                            |
|---------|------------|-----------|--------------------------------------------------------------------------------------------------------------------------------------------|
| D15–D12 | Reserved   |           | These bits are undefined when read.                                                                                                        |
| D11–D0  | JAB[11..0] | Read Only | The respective port's JAB bit is set to 1 when the Jabber condition is detected on that port. After reset, these bits are cleared to zero. |

### 4.7 ADMINISTRATION REGISTER (ADMIN)

Page 0 Address 4h

| Bit     | Bit Name                          | Access | Bit Description                                                                                                                                                                                                                                                                                                                                                                                              |
|---------|-----------------------------------|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D15–D12 | Reserved                          |        | For compatibility with future enhanced versions these bits must be written as zero. They are undefined when read.                                                                                                                                                                                                                                                                                            |
| D11–D0  | ADMIN_DIS[11] ...<br>ADMIN_DIS[0] | R/W    | Setting these bits to 0 enables the respective port (TX and RX). Writing a 1 to any bit will disable that port. Note that port enable/disable actions will occur at the next network idle period. For example, if an ADMIN_DIS bit is cleared during an incoming packet, this port will only be enabled after the incoming packet has finished. After reset, these bits default to zero (all ports enabled). |

## 4.0 Registers (Continued)

### 4.8 DEVICE ID REGISTER (DEVICEID)

Page 0 Address 5h

| Bit     | Bit Name | Access    | Bit Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|---------|----------|-----------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D15–D12 | Reserved |           | For compatibility with future enhanced versions these bits must be written as zero. They are undefined when read.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| D11–D8  | PORT_NUM | Read Only | Port Number: These bits indicate the last or current receiving port number.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| D7      | EE_DONE  | Read Only | EEPROM Access Done: This bit is set when the DP83850 has completed its read of the EEPROM.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| D6      | Reserved |           | For compatibility with future enhanced versions these bits must be written as zero. They are undefined when read.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| D5      | RID_ER   | R/W       | Repeater ID Error: This bit is set under two conditions: <ol style="list-style-type: none"> <li>1. When this DP83850 sees another DP83850 use the same RID number as its own on the management bus, or,</li> <li>2. RID[4:0] has been programmed with a value of 1Fh.</li> </ol> This bit sticks to 1 until it is cleared by a register write.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| D4–D0   | RPTR_ID  | R/W       | Device ID: These bits are the source for the IR_VECT[4:0] pins. These bits also supply the register address for MII serial bus accesses. At the rising edge of $\overline{RST}$ , the levels on RID[4:0] are latched in this register as D[4:0]. <p><b>Note 1:</b> While you can write to these bits at any time, caution must be used. First, when a new value is entered, all subsequent accesses must be performed at this new address. Second, if an RID number is chosen that is the same as another DP83850 device, both of these devices will be rendered unreadable (there will be contention). Recovery from this condition is only possible with a complete system reset, since it will not be possible to write new unique RIDs to the contending DP83850s.</p> <p><b>Note 2:</b> Since IR_VECT = 1Fh is an illegal value, D[4:0] must not be written to this value.</p> |

### 4.9 HUB ID 0 REGISTER (HUBID0)

Page 0 Address 6h

| Bit    | Bit Name      | Access | Bit Description                                                                                                                               |
|--------|---------------|--------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| D15–D0 | HUB_ID0[15:0] | R/W    | Hub ID0: Contains the first 16 bits read from the EEPROM. The first bit read will be written to HUB_ID0[0]; the last bit read to HUB_ID0[15]. |

### 4.10 HUB ID 1 REGISTER (HUBID1)

Page 0 Address 7h

| Bit    | Bit Name      | Access | Bit Description                                                                                                                                |
|--------|---------------|--------|------------------------------------------------------------------------------------------------------------------------------------------------|
| D15–D0 | HUB_ID1[15:0] | R/W    | Hub ID1: Contains the second 16 bits read from the EEPROM. The first bit read will be written to HUB_ID1[0]; the last bit read to HUB_ID1[15]. |

## 4.0 Registers (Continued)

### 4.11 PORT MANAGEMENT COUNTER REGISTERS

Each of the 12 ports of the DP83850 has a set of 4 event counters whose values can be read or pre-set (written) through the Port Management Counter Registers. Ports 0 through 5 have their registers in register page 0 and ports 6 through 11 in register page 1.

All counters will rollover to zero after reaching their maximum count: they are not “sticky”. There is no interrupt on reaching maximum count, so the management software must ensure the registers are polled often enough so as not to rollover twice; management software can deduce a single rollover as long as the counter has not yet reached the previously read value (a simple compare). It is safest for the management software to guarantee to check all counters at least once per possible rollover time. All counters are cleared to zero at power-on and/or reset ( $\overline{RST}$  asserted).

The Short Event, Late Event and Collision Counters are 32 bits long. Since the corresponding Counter Registers are only 16 bits, the DP83850 has to internally multiplex the counter value into two 16-bit values that the management software must then concatenate to form the full 32-bit value. Some restrictions apply to the access of the counter registers:

1. A 32-bit counter must be read as two consecutive 16-bit accesses. Upon the first access, the DP83850 places the full 32-bit counter value in a holding register, from where it transfers the upper 16 bits first. The second access reads the lower 16 bits of the counter. If there is any access to another register in between the counter reads, the concatenated value of the counter will be invalid (the DP83850’s internal multiplexer will reset).
2. For the same reason, a 32-bit counter must be written as two consecutive 16-bit accesses.
3. All counters are cleared by writing 0000 0000h to them. The counter value is unaffected by read accesses.
4. The counters should only be written to when they are disabled. This is done by deasserting the MGTEN bit in the CONFIG register.
5. No 32-bit counter register may be read or written to continuously. A counter can only be accessed again after an access to a different counter. For 32-bit counters, once the two consecutive 16-bit accesses are done, the next counter register access must be to a different counter register. In practice this is not a burden: Management software should periodically gather counter statistics from all ports, all counters, in a sequential manner.

#### 4.11.1 Short Event Counter Registers

Per port (“n” = port number) counters that indicate the number of Carrier Events that were active for less than the ShortEventMaxTime, which is defined as between 74 and 82 (76 nominal) bit times.

| Bit    | Access | Bit Description                                                                                  |
|--------|--------|--------------------------------------------------------------------------------------------------|
| D15-D0 | R/W    | First Access—Most Significant Word of P“n”_SE<br>Second Access—Least Significant Word of P“n”_SE |

#### 4.11.2 Late Event Counter Registers

Per port (“n” = port number) counters that indicate the number of collisions that occurred after the LateEventThreshold, which is defined to be 480 to 565 bit times (512 nominal). Both the Late Event and Collision Counters will be incremented when this event occurs.

| Bit    | Access | Bit Description                                                                                  |
|--------|--------|--------------------------------------------------------------------------------------------------|
| D15-D0 | R/W    | First Access—Most Significant Word of P“n”_LE<br>Second Access—Least Significant Word of P“n”_LE |

**4.11.3 Collision Counter Registers**

Per port ("n" = port number) counters that indicate the number of collisions (COL asserted).

| Bit    | Access | Bit Description                                                                                      |
|--------|--------|------------------------------------------------------------------------------------------------------|
| D15-D0 | R/W    | First Access—Most Significant Word of P“n”__COL<br>Second Access—Least Significant Word of P“n”__COL |

**4.11.4 Auto-Partition Counter Registers**

Per port ("n" = port number) counters that indicate the number of times the port was auto-partitioned.

| Bit    | Access | Bit Description |
|--------|--------|-----------------|
| D15-D0 | R/W    | P“n”__PART      |

**4.12 SILICON REVISION REGISTER (SIREV)**

Page 1      Address 4h

| Bit    | Bit Name      | Access    | Bit Description                           |
|--------|---------------|-----------|-------------------------------------------|
| D15-D0 | SI__REV[15:0] | Read Only | Silicon Revision: Currently Reads All 0's |

## 5.0 DP83850 Applications

### 5.1 MII INTERFACE CONNECTIONS

The DP83850's interface to DP83840 PHY devices is fully described in AN1010 "100BASE-TX Unmanaged Repeater Design Recommendations". Designers should be aware that there are significant issues involved in the signal timing, loading and layout of this interface and they should consult

AN1010 and/or their National Semiconductor representative prior to attempting a design. Further system timing analysis shows that the RXD[3:0], RX\_DV and RX\_ER signals should be latched into the DP83850 from the connected DP83840s. *Figure 3* shows the recommended scheme. This ensures system timing can be met for hub stacks.

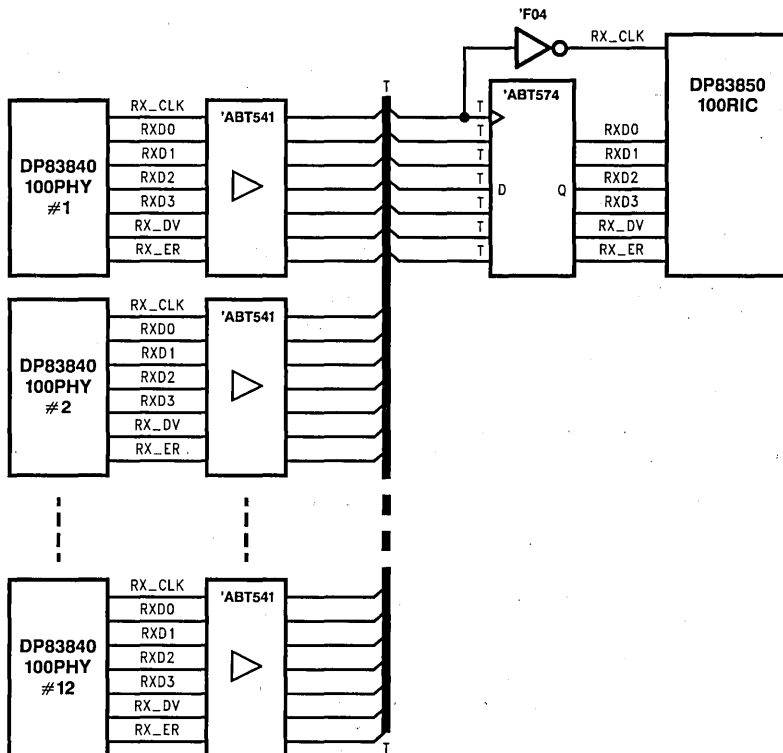


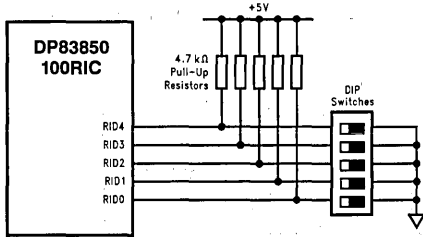
FIGURE 3. Recommended DP83840 to DP83850 Connections

TL/F/12391-16

## 5.0 DP83850 Applications (Continued)

### 5.2 REPEATER ID INTERFACE

The repeater ID interface is shown in *Figure 4*. It consists of a bank of DIP switches or links to set the RID number for the DP83850 to use as its IR\_VECT[4:0] number.



TL/F/12391-17

FIGURE 4. DP83850 Repeater ID Number Interface

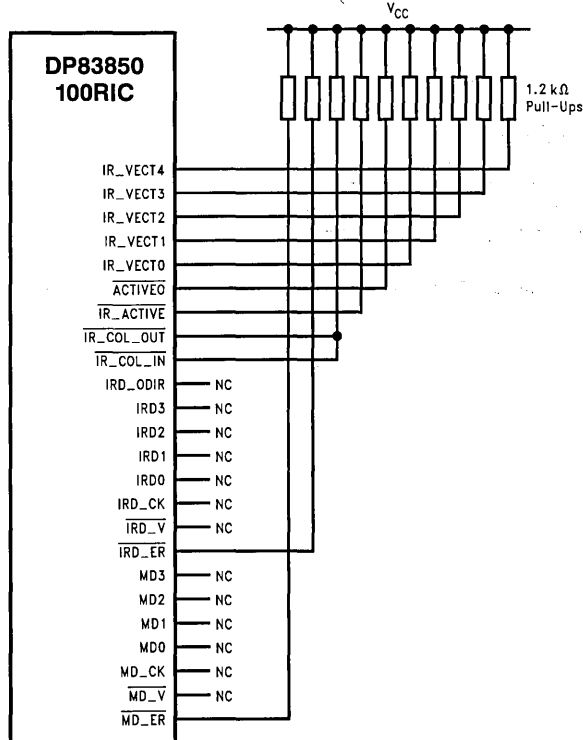
### 5.3 INTER REPEATER BUS CONNECTIONS

For a simple standalone repeater that cannot be stacked, no inter repeater bus transceivers/drivers are required and the inter repeater bus interface is simple. An example of this is shown in *Figure 5*.

For a stackable hub design, the DP83850's Inter Repeater Bus connections are complex and have many issues regarding signal timing, loading and layout. An example design for a TTL level inter repeater bus is given in *Figure 6*. It should be noted that this is just one way of connecting up an inter repeater bus. There are many other possible ways to design this interface. Designers should be aware that tim-

ing, particularly skew between clock and data, is critical. For this reason it is not recommended to use LS, S, TTL, or CMOS logic drivers. The ABT family of logic is recommended, the FAST family could possibly be made to work too. Also recommended is the BTL logic transceiver family; this approach has the advantage of significantly lower noise and would make passing of FCC and other EMI tests easier.

In *Figure 6*, one of the "spare" 74ABT125C drivers is used as a delay element for the 74ABT244 in the IR\_VECT scheme: this reduces turnaround bus contention. Alternatively, one of the spare 74AC32 gates from the LED logic could be used for this purpose. Similarly, the 74F27 gate reduces bus contention and decodes the correct output enable conditions for the 74ABT16245C bus transceiver. The pull up resistors on the DP83850 should be a minimum of 1.2 kΩ. Lower values may be required depending on layout/loading, especially on the ACTIVE0 and IR\_ACTIVE signals where short deassertion time is critical. The value of the pull up resistor terminations on the inter repeater bus backplane will depend on the bus loading. The values should be chosen so that the signals on the bus have fast enough edges to meet the DP83850 inter repeater bus timings. The inter repeater bus will need to be terminated properly at each end to prevent signal reflections from causing problems. The IRD\_CK and M\_CK signal levels are adjusted with resistors in order to achieve a better mark to space ratio over the inter repeater bus; in each case, the 390Ω and 600Ω attenuator needs to be placed physically close to the 100RIC while the 22Ω resistor needs to be placed physically close to the 'ABT16245C transceiver.



TL/F/12391-18

FIGURE 5. DP83850 Standalone Inter Repeater Bus Interface



5.0 DP83850 Applications (Continued)

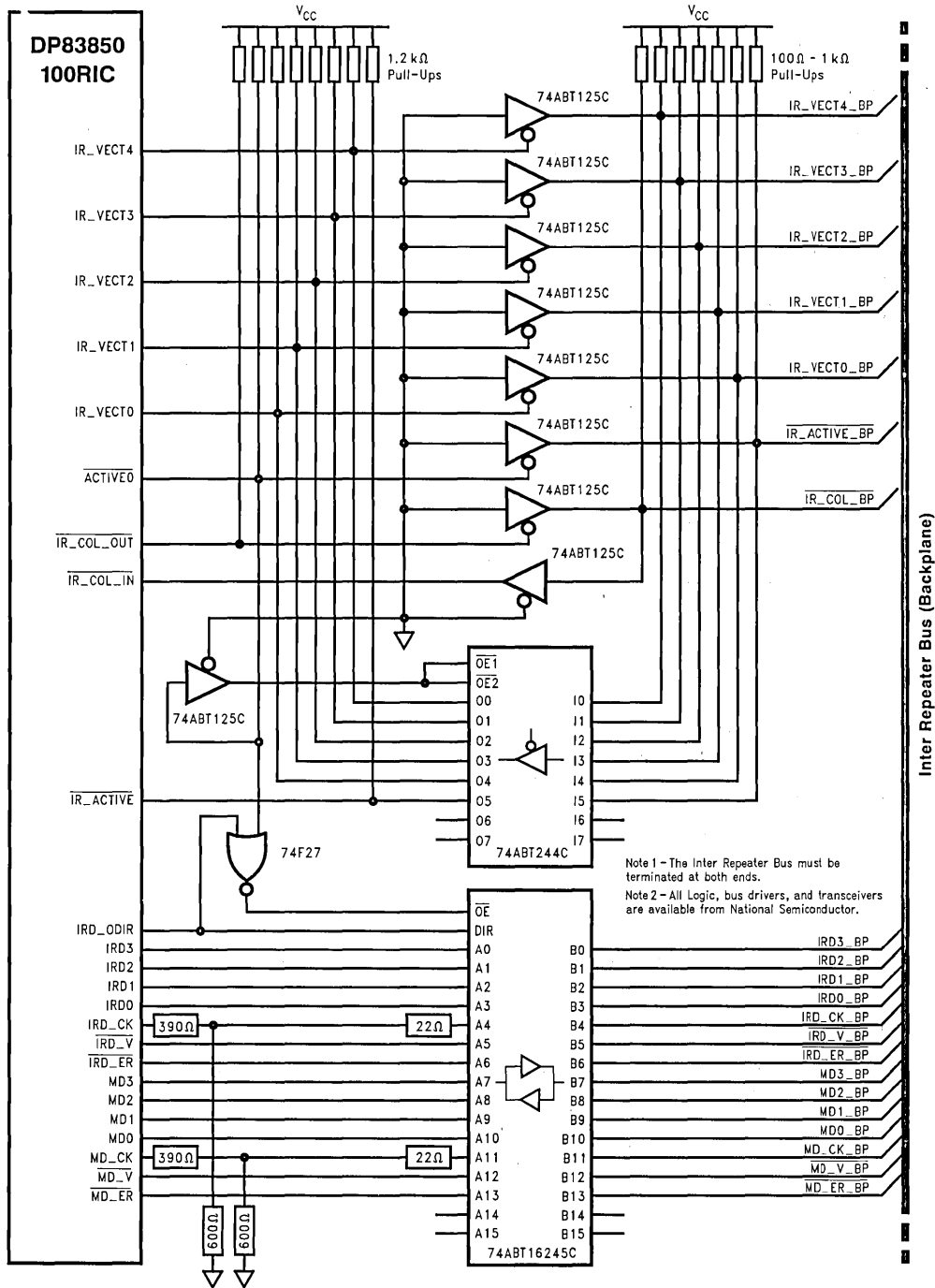


FIGURE 6. Inter Repeater Bus Connections

TL/F/12391-19

## 5.0 DP83850 Applications (Continued)

### 5.4 DP83856 100RIB CONNECTIONS

To achieve a practical managed 100 Mb/s repeater design that keeps up with the fast flow of network information, a hardware statistics gathering engine is required. The DP83856 100 Mb/s Repeater Information Base device (100RIB) is specifically designed to work with the DP83856 to provide such a design. In a multi-100RIC system, one of the 100RIC devices has to be chosen to source the transmit data bus to the 100RIB. This 100RIC is known as the

"Local 100RIC" since it is likely to be the nearest one (physically) to the 100RIB on the circuit board. All the other signals that the 100RIB requires in order to keep statistics are common to all the other 100RICs. Figure 7 shows a typical connection between the 100RIC and the 100RIB. Note that, depending on board layout, track lengths and loading, buffers (not shown) may be required on some signals.

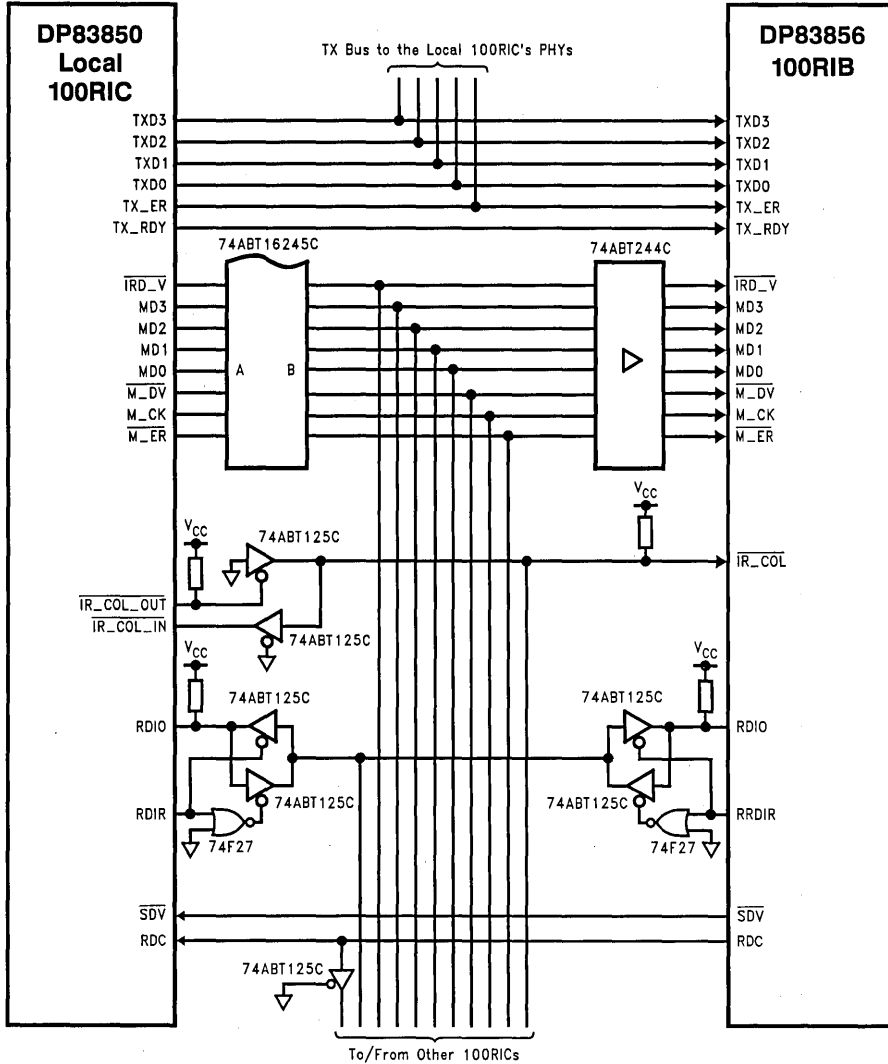


FIGURE 7. Typical DP83850 to DP83856 Connections

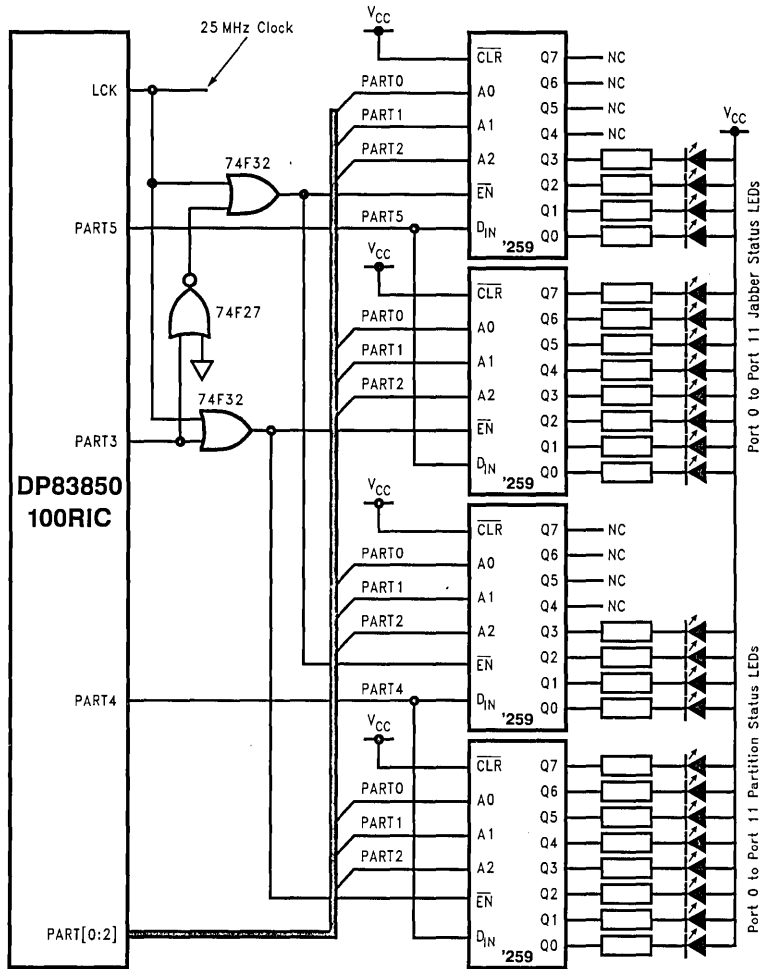
TL/F/12391-20

### 5.0 DP83850 Applications (Continued)

#### 5.5 PORT PARTITION AND JABBER STATUS LEDs

Port Partition and Jabber Status must be decoded from the PART[5:0] outputs as described in Section 3.11. One possible decoder implementation is shown in Figure 8. This uses '259 addressable latches to hold the LED status for each

port. The lowest significant 3 bits of the port address (PART[2:0]) are directly connected to each of the '259 addressable latches. The most significant address bit (PART3) and its inverse are gated by the system clock to produce low going pulses to the '259 enables at the correct time.



TL/F/12391-21

FIGURE 8. Implementation of a Jabber and Partition Status LED Scheme

## 6.0 AC and DC Specifications

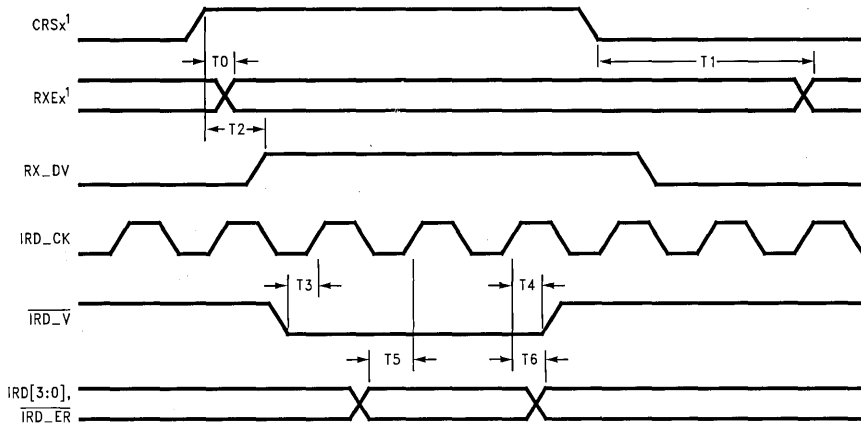
### 6.1 DC SPECIFICATIONS

| Symbol   | Parameter                         | Conditions              | Min | Max       | Units   |
|----------|-----------------------------------|-------------------------|-----|-----------|---------|
| $V_{OH}$ | Minimum High Level Output Voltage |                         | 3.7 |           | V       |
| $V_{OL}$ | Minimum Low Level Output Voltage  | $I_{OL} = 1$ to 4 mA    |     | 0.5       | V       |
| $V_{IH}$ | Minimum High Level Input Voltage  | TTL Input               | 2.0 |           | V       |
| $V_{IL}$ | Maximum Low Level Input Voltage   | TTL Input               |     | 0.8       | V       |
| $I_{IN}$ | Input Current                     | With Internal Pullup    |     | $\pm 150$ | $\mu A$ |
|          |                                   | Without Internal Pullup |     | $\pm 10$  |         |
| $I_{OL}$ | Maximum Low Level Output Current  | TXD Outputs             |     | 24        | mA      |
|          |                                   | IR Bus Outputs          |     | 4         |         |
|          |                                   | TXER Output             |     | 12        |         |
| $I_{OZ}$ | TRI-STATE Output Leakage Current  |                         |     | $\pm 10$  | $\mu A$ |
| $I_{CC}$ | Average Supply Current            | Typical                 |     | 260       | mA      |

### 6.2 AC SPECIFICATIONS

**Note:** All AC timings given are based on calculations. They are not the result of device characterization so cannot be guaranteed over temperature or supply voltage variations.

#### 6.2.1 Receive Timing



TL/F/12391-5

| Parameter | Description                                            | Min | Max | Units |
|-----------|--------------------------------------------------------|-----|-----|-------|
| T0        | CRSx to RXEx Assertion Delay                           |     | 20  | ns    |
| T1        | CRSx to RXEx De-Assertion Delay with no Collision      | 3   | 5   | LCK   |
| T2        | CRSx to RX_DV Delay Requirement (Note 2)               | 1   | 4   | RXC   |
| T3        | $\overline{IRD\_V}$ Setup to IRD_CK High               | 5   |     | ns    |
| T4        | $\overline{IRD\_V}$ Hold from IRD_CK High              | 5   |     | ns    |
| T5        | IRD[3:0] or $\overline{IRD\_ER}$ Setup to IRD_CK High  | 5   |     | ns    |
| T6        | IRD[3:0] or $\overline{IRD\_ER}$ Hold from IRD_CK High | 5   |     | ns    |

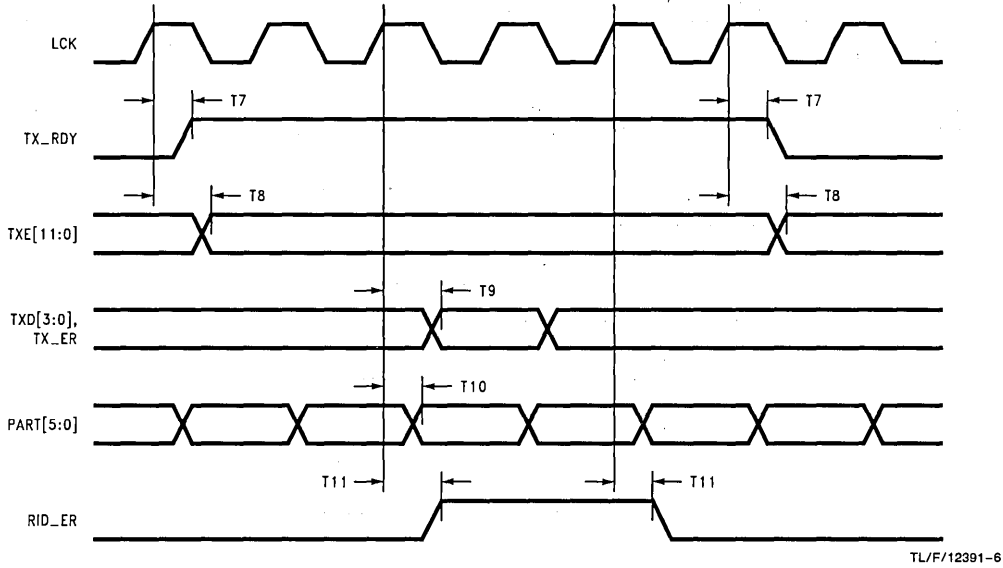
**Note 1:** "CRSx" and "RXEx" refer to any of the CRS[11:0] or RXE[11:0] signals. In the event of a collision (more than one CRS is active) none of the RXE signals will be asserted.

**Note 2:** If, after 4 RXC clocks from CSRx going high, no aligned data is received, the DP83850 100R1C will repeat the JAM pattern.

## 6.0 AC and DC Specifications (Continued)

### 6.2.2 Transmit, Partition, and RID\_ER Timings

| Parameter | Description                                | Min | Max | Units |
|-----------|--------------------------------------------|-----|-----|-------|
| T7        | TX_RDY Delay from LCK High                 | 7   | 33  | ns    |
| T8        | TXE[11:0] Delay from LCK High              | 7   | 30  | ns    |
| T9        | TXD[3:0] or TX_ER Valid Time from LCK High | 7   | 30  | ns    |
| T10       | PART[5:0] Valid Time from LCK High         |     | 25  | ns    |
| T11       | RID_ER Delay from LCK High                 |     | 25  | ns    |



## 6.0 AC and DC Specifications (Continued)

### 6.2.3 Inter Repeater Receive and Intra-Repeater Collision

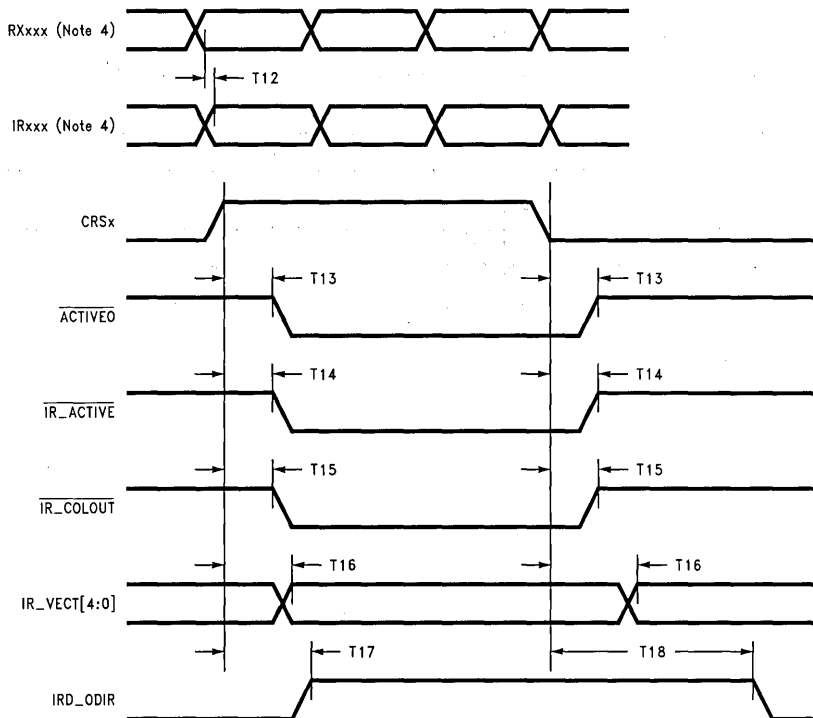
| Parameter | Description                                                                                           | Min | Max | Units |
|-----------|-------------------------------------------------------------------------------------------------------|-----|-----|-------|
| T12       | Receive to Inter Repeater Bus Delay (Note 3)                                                          |     | 11  | ns    |
| T13       | CRSx Assertion (De-Assertion) to $\overline{\text{ACTIVE0}}$ Assertion (De-Assertion) (Note 4)        |     | 22  | ns    |
| T14       | CRSx Assertion (De-Assertion) to $\overline{\text{IR\_ACTIVE}}$ Assertion (De-Assertion) (Note 4)     |     | 22  | ns    |
| T15       | CRSx Assertion (De-Assertion) to $\overline{\text{IR\_COLOUT}}$ Assertion (De-Assertion) (Notes 4, 5) |     | 22  | ns    |
| T16       | CRSx Assertion (De-Assertion) to $\overline{\text{IR\_VECT[4:0]}}$ Assertion (De-Assertion) (Note 4)  |     | 24  | ns    |
| T17       | CRSx Assertion to $\overline{\text{IRD\_ODIR}}$ Assertion (Note 4)                                    |     | 42  | ns    |
| T18       | CRSx De-Assertion to $\overline{\text{IRD\_ODIR}}$ De-Assertion (Notes 4, 6)                          | 4   | 6   | LCK   |

**Note 3:** "RXxxx" refers to any of the receive signals, i.e. RXC, RXD[3:0], RX\_DV, or RX\_ER. "IRxxx" refers to any of the Inter Repeater signals, i.e. IRD\_CK, IRD[3:0], IRD\_V, or IRD\_ER.

**Note 4:** "CRSx" refers to any of CRS[11:0] signals being asserted.

**Note 5:** This timing refers to the assertion of  $\overline{\text{IR\_COL\_OUT}}$  during an internal collision, that is when 2 or more CRSx signals are asserted in the same DP83850.

**Note 6:** This timing refers only to the condition where only one CRSx is present.  $\overline{\text{IRD\_ODIR}}$  will be de-asserted immediately if a collision occurs.



TL/F/12391-7

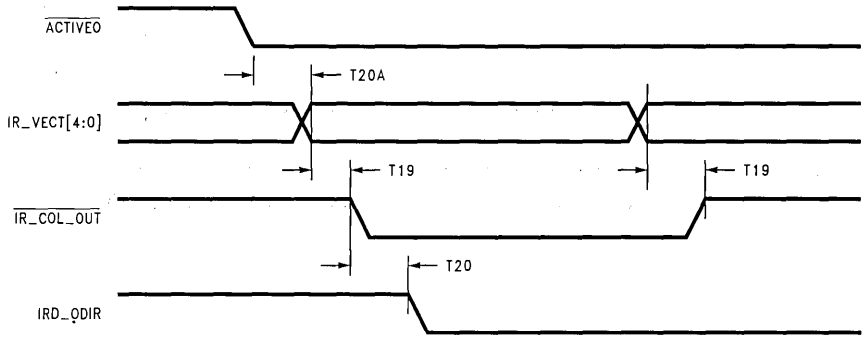
## 6.0 AC and DC Specifications (Continued)

### 6.2.4 Inter Repeater Collision

| Parameter | Description                                                         | Min | Max | Units |
|-----------|---------------------------------------------------------------------|-----|-----|-------|
| T19       | IR_VECT[4:0] Change to IR_COL_OUT Assertion (De-Assertion) (Note 7) |     | 22  | ns    |
| T20       | IR_COL_OUT Assertion to IRD_ODIR De-Assertion                       |     | 20  | ns    |
| T20A      | ACTIVE0 Low to IR_VECT[4:0] Feedback (Note 8)                       |     | 22  | ns    |

**Note 7:** This timing refers to the condition where the repeater has detected a change from its driven arbitration vector to what is seen on the IR\_VECT[4:0] bus. In other words, an "Inter Repeater" collision is occurring.

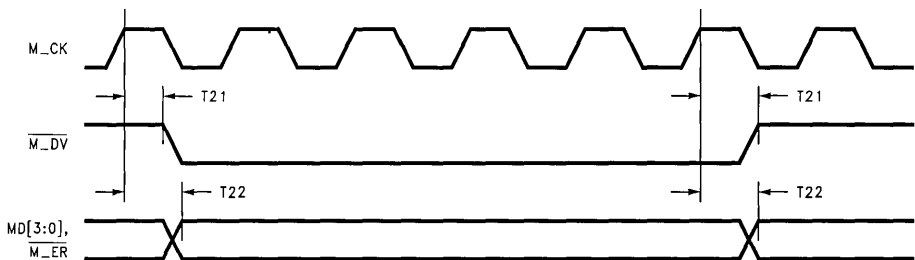
**Note 8:** This timing refers to the condition where the DP83850 first drives its vector onto IR\_VECT[4:0] at the beginning of a packet. The IR\_VECT[4:0] feedback (possibly returning from an external bus) must be stable by this time.



TL/F/12391-8

### 6.2.5 Management Bus—Output Mode

| Parameter | Description                                  | Min | Max | Units |
|-----------|----------------------------------------------|-----|-----|-------|
| T21       | M_DV Assertion (De-Assertion) from M_CK High | 3   | 13  | ns    |
| T22       | MD[3:0] or M_ER Valid from M_CK High         | 3   | 13  | ns    |
| T23       | Removed                                      |     |     |       |

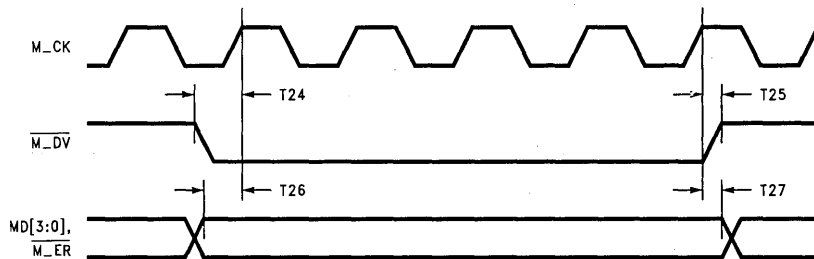


TL/F/12391-9

## 6.0 AC and DC Specifications (Continued)

### 6.2.6 Management Bus—Input Mode

| Parameter | Description                                            | Min | Max | Units |
|-----------|--------------------------------------------------------|-----|-----|-------|
| T24       | $\overline{M\_DV}$ Setup to $M\_CK$ High               | 5   |     | ns    |
| T25       | $\overline{M\_DV}$ Hold from $M\_CK$ High              | 5   |     | ns    |
| T26       | $MD[3:0]$ or $\overline{M\_ER}$ Setup to $M\_CK$ High  | 5   |     | ns    |
| T27       | $MD[3:0]$ or $\overline{M\_ER}$ Hold from $M\_CK$ High | 5   |     | ns    |



TL/F/12391-10



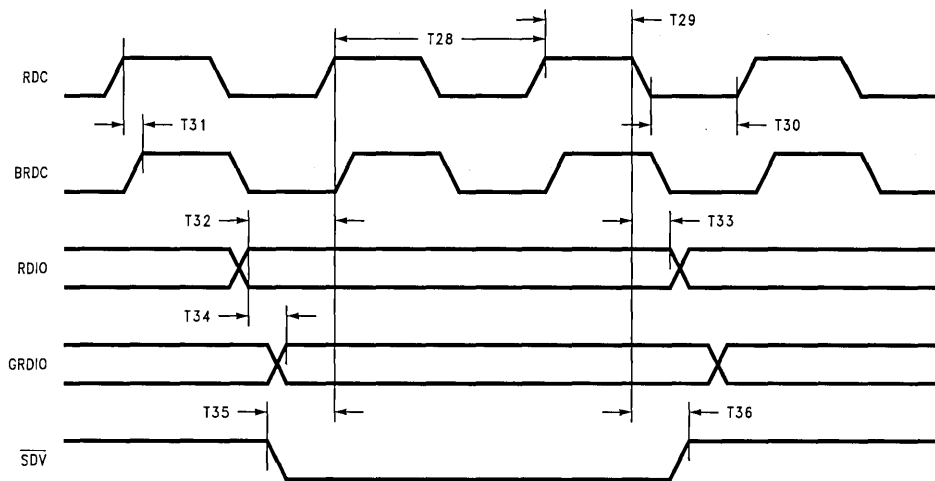
## 6.0 AC and DC Specifications (Continued)

### 6.2.7 Serial Register Write

| Parameter | Description                         | Min | Max | Units |
|-----------|-------------------------------------|-----|-----|-------|
| T28       | RDC Period                          | 400 |     | ns    |
| T29       | RDC High Time (Note 9)              | 40  |     | ns    |
| T30       | RDC Low Time (Note 9)               | 40  |     | ns    |
| T31       | RDC to BRDC Delay                   |     | 25  | ns    |
| T32       | RDIO Setup to RDC High              | 10  |     | ns    |
| T33       | RDIO Hold from RDC High             | 10  |     | ns    |
| T34       | RDIO to GRDIO Delay (Note 10)       |     | 25  | ns    |
| T35       | $\overline{SDV}$ Setup to RDC High  | 10  |     | ns    |
| T36       | $\overline{SDV}$ Hold from RDC High | 10  |     | ns    |

**Note 9:** Although the High or Low time may be as small as 40 ns, the RDC cycle time is limited to 2.5 MHz max.

**Note 10:** Serial data will be gated from RDIO to GRDIO during write operations when the "phy\_access" bit in the CONFIG register is set.



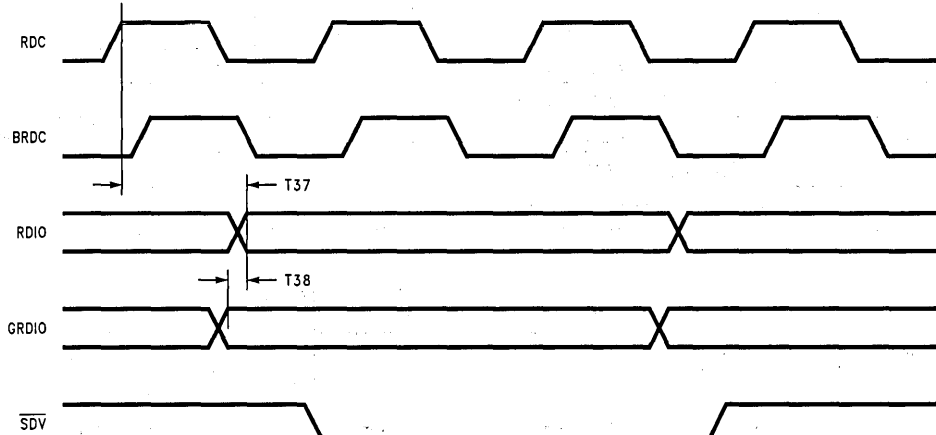
TL/F/12391-11

## 6.0 AC and DC Specifications (Continued)

### 6.2.8 Serial Register Read

| Parameter | Description                   | Min | Max | Units |
|-----------|-------------------------------|-----|-----|-------|
| T37       | RDIO Valid from RDC           |     | 25  | ns    |
| T38       | GRDIO to RDIO Delay (Note 11) |     | 25  | ns    |

**Note 11:** Serial data will be gated from GRDIO to RDIO during read operations when the "phy\_access" bit in the CONFIG register is set.



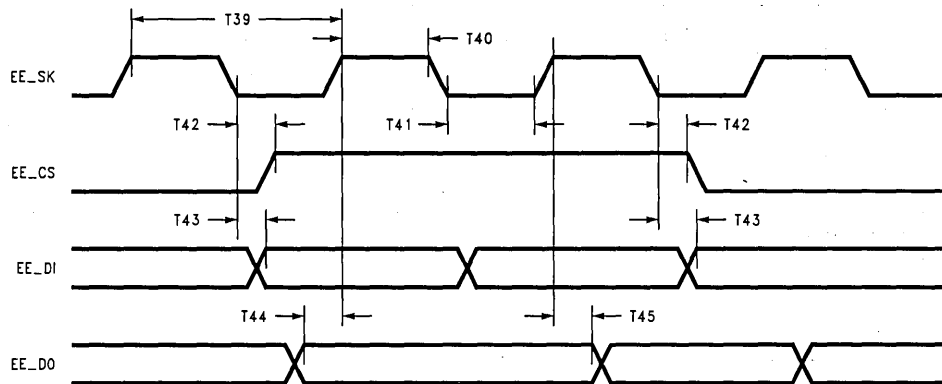
TL/F/12391-12

## 6.0 AC and DC Specifications (Continued)

### 6.2.9. EEPROM Access

| Parameter | Description                                   | Min | Max        | Units |
|-----------|-----------------------------------------------|-----|------------|-------|
| T39       | EE_SK Period (Note 12)                        |     | 1280 (Nom) | ns    |
| T40       | EE_SK High Time (Note 12)                     |     | 640 (Nom)  | ns    |
| T41       | EE_SK Low Time (Note 12)                      |     | 640 (Nom)  | ns    |
| T42       | EE_CS Assertion (De-Assertion) from EE_SK Low |     | 25         | ns    |
| T43       | EE_DI Assertion (De-Assertion) from EE_SK Low |     | 25         | ns    |
| T44       | EE_DO Setup to EE_SK High                     |     | 10         | ns    |
| T45       | EE_DO Hold from EE_SK High                    |     | 0          | ns    |

Note 12: These timings are nominal (untested) values.



TL/F/12391-13

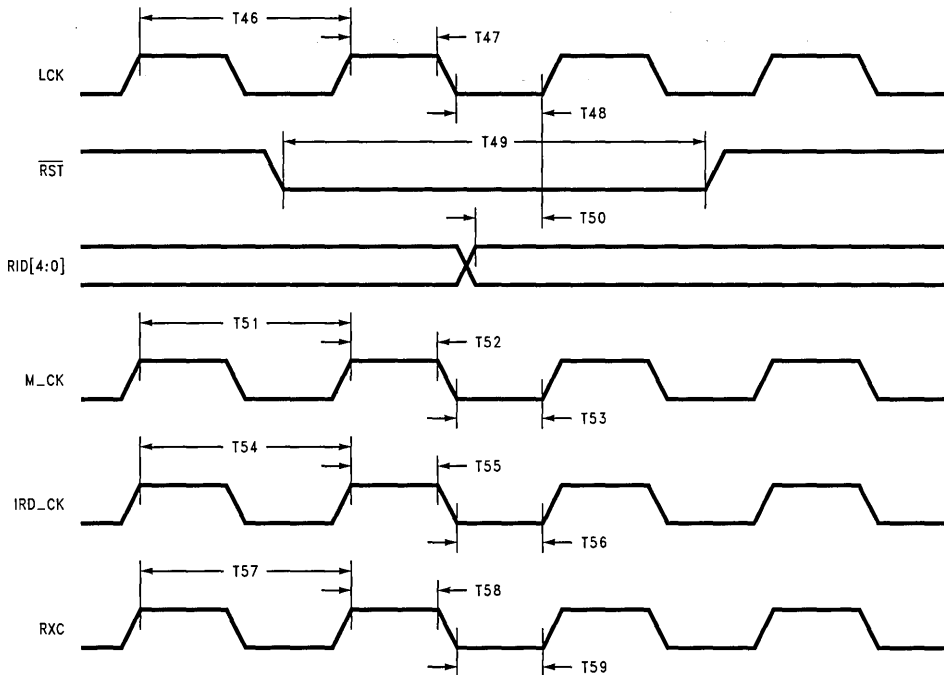
## 6.0 AC and DC Specifications (Continued)

### 6.2.10. Clocks, Reset, RID

| Parameter | Description                            | Min | Max      | Units |
|-----------|----------------------------------------|-----|----------|-------|
| T46       | LCK Period                             | 40  | 40       | ns    |
| T47       | LCK High Time                          | 16  |          | ns    |
| T48       | LCK Low Time                           | 16  |          | ns    |
| T49       | $\overline{\text{RST}}$ Assertion Time | 75  |          | LCK   |
| T50       | RID[4:0] Setup to LCK High             | 20  |          | ns    |
| T51       | M_CK Period                            | 40  | 40       | ns    |
| T52       | M_CK High Time                         | 16  |          | ns    |
| T53       | M_CK Low Time                          | 16  |          | ns    |
| T54       | IRD_CK Period                          | 40  | 40       | ns    |
| T55       | IRD_CK High Time                       | 16  |          | ns    |
| T56       | IRD_CK Low Time                        | 16  |          | ns    |
| T57       | RXC Period                             | 40  |          | ns    |
| T58       | RXC High Time                          | 16  |          | ns    |
| T59       | RXC Low Time                           | 16  |          | ns    |
| T60       | xCK Frequency Tolerance (Notes 13, 14) |     | $\pm 50$ | ppm   |

**Note 13:** Frequency tolerance is respect to an ideal 25 MHz source.

**Note 14:** "xCK" refers to any of LCK, IRD\_CK or RXC.



TL/F/12391-14

# DP83856 100 Mb/s Repeater Information Base

## General Description

The DP83856 100 Mb/s Repeater Information Base is designed specifically to meet the management demands of today's high speed Ethernet networking systems.

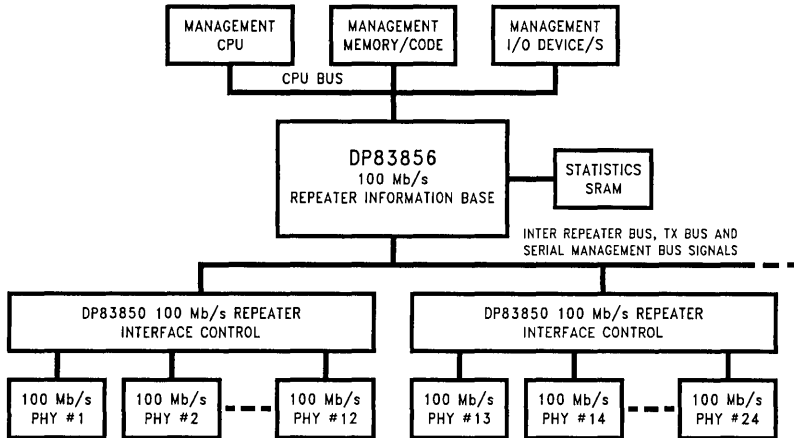
The DP83856 simplifies design of managed multiport repeaters. Used in conjunction with up to 16 DP83850s it enables a repeater system to become a single managed entity that is fully compatible with the IEEE 802.3u clause 30 management requirements.

The DP83856 device incorporates all the necessary functions and counters for collecting network statistics. Information is gathered on a per-packet, per-port basis: the port which is receiving the packet is the active port for statistics collection.

## Features

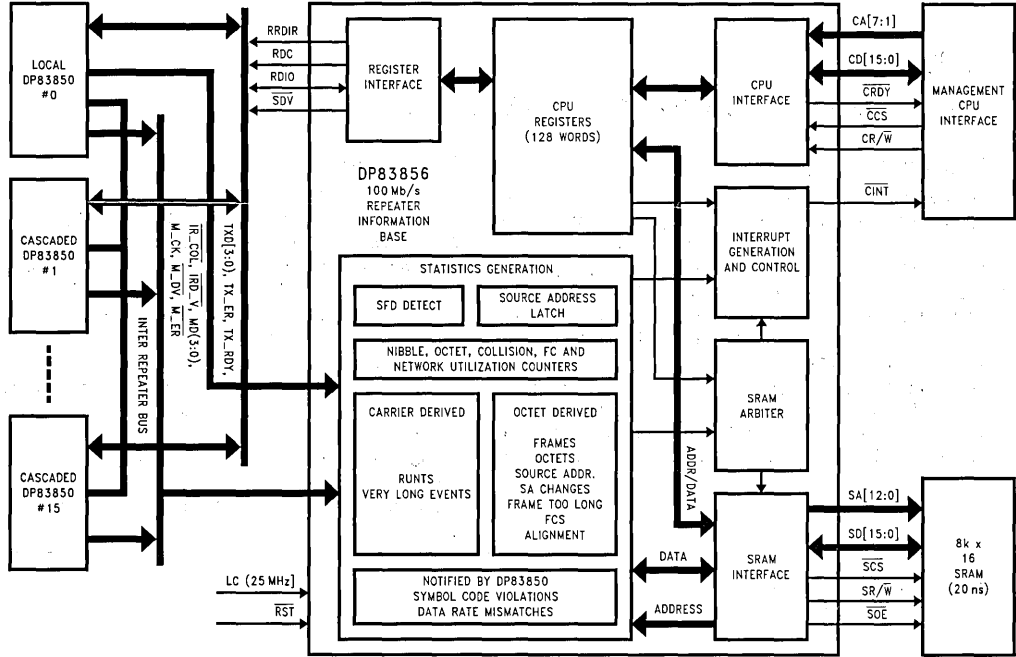
- Supports up to 16 DP83850 Repeater Interface Controllers (192, 100 Mb/s ports on one segment)
- Fully IEEE 802.3u clause 30 compatible
- Network management statistics processed on a per activity (per packet) basis
- Programmed I/O interface for statistics reporting
- Uses external SRAM to maintain per port network management statistics counters
- Single interrupt acknowledgement provides report on all per port SRAM based and DP83856 based statistics
- Parallel register interface to CPU (16-bit)
- Allows indirect access to the DP83850 Repeater Interface Controller and DP83840 Physical Layer Device serial registers through a parallel register interface
- 132-pin PQFP

## System Diagram



TL/F/12392-1

# Block Diagram



TL/F/12392-2

## Table of Contents

### 1.0 PIN CONNECTION DIAGRAM

### 2.0 PIN DESCRIPTION

- 2.1 CPU Interface
- 2.2 SRAM Interface
- 2.3 Transmit Bus and Management Bus
- 2.4 MII Interface
- 2.5 Test Interface
- 2.6 Miscellaneous Pins
- 2.7 Pin Type Designation

### 3.0 FUNCTIONAL DESCRIPTION

- 3.1 Statistics Generation
- 3.2 SRAM Interface
- 3.3 SRAM Arbiter
- 3.4 Interrupt Generation and Control
- 3.5 MII Register Interface
- 3.6 CPU Register Block
- 3.7 Management Data Interface

### 4.0 REGISTERS

- 4.1 Register Memory Map
- 4.2 Configuration Register
- 4.3 Interrupt Register
- 4.4 SRAM Interface Register
- 4.5 MII Management Interface Register
- 4.6 SRAM Write Data Register
- 4.7 MII Write Data Register
- 4.8 Device ID Register
- 4.9 SRAM Read Data Registers
- 4.10 Carrier Count Register
- 4.11 Oct\_Nib Count Register
- 4.12 Network Counters
- 4.13 MII Read Data Registers

### 5.0 DP83956 Initialization

- 5.1 SRAM Test and Initialization
- 5.2 Network Counter Initialization
- 5.3 Interrupt Initialization and Usage

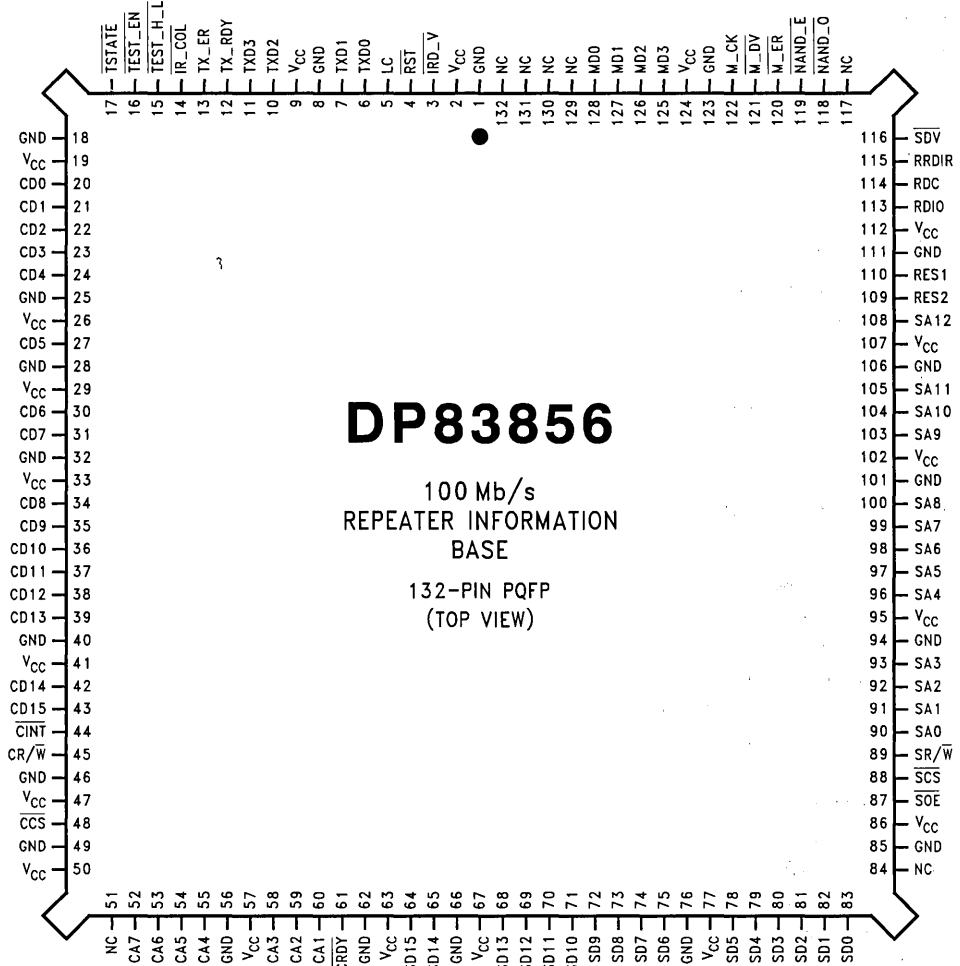
### 6.0 Repeater MIB Support

- 6.1 Repeater MIB Support

### 7.0 AC AND DC SPECIFICATIONS

- 7.1 DC Specifications
- 7.2 AC Specifications

# 1.0 Pin Connection Diagram



TL/F/12392-3



## 2.0 Pin Description

### 2.1 CPU INTERFACE

The CPU interface pins are a set of generic interface signals designed to accommodate many different CPU types with minimal external logic. The data interface is 16-bits wide and does not provide any steering capabilities. Furthermore, all accesses must be aligned on 16-bit boundaries, as indicated in the CPU Register Map Section 4.

| Signal Name                     | Type     | Active | Description                                                                                                                                                                                                                                                                                                                        |
|---------------------------------|----------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| $\overline{\text{CINT}}$        | O/Z, L   | Low    | <b>CPU INTERRUPT:</b> Indicates that the DP83856 has at least one interrupt pending. The $\overline{\text{CINT}}$ signal will remain active until the CPU reads the Interrupt Register. It is software's responsibility to keep track of multiple interrupts pending, and service all of the interrupts.                           |
| $\overline{\text{CRDY}}$        | O/Z, L   | Low    | <b>CPU READY:</b> Indicates that the DP83856 is ready to terminate the current cycle. The DP83856 asserts $\overline{\text{CRDY}}$ on writes once it has strobed the data into its write data holding register. The DP83856 asserts $\overline{\text{CRDY}}$ on reads once it has strobed data into its read data output register. |
| $\overline{\text{CCS}}$         | I        | Low    | <b>CPU CHIP SELECT:</b> Chip select for internal DP83856 registers. Generated by external logic as an address decode of the DP83856 register space. $\overline{\text{CCS}}$ must remain valid for the entire cycle.                                                                                                                |
| $\text{CR}/\overline{\text{W}}$ | I        |        | <b>CPU READ-WRITE:</b> Read/Write strobe for DP83856 internal registers.<br>Read = 1, Write = 0.                                                                                                                                                                                                                                   |
| CA[7:1]                         | I        |        | <b>CPU ADDRESS [7:1]:</b> Address bus for DP83856 register accesses. The DP83856 latches the address for internal use within 45 ns of $\overline{\text{CCS}}$ being asserted.                                                                                                                                                      |
| CD[15:0]                        | I/O/Z, M |        | <b>CPU DATA [15:0]:</b> 16-bit data bus for DP83856 register accesses. CD[15:0] correspond to the low 16-bits of data on the CPU. The DP83856 implements <b>Big Endian</b> convention for data storage. All CPU register accesses should be 16-bit accesses aligned on 16-bit boundaries.                                          |

### 2.2 SRAM INTERFACE

The SRAM interface pins are used to connect the DP83856 to a fast (20 ns) external SRAM. The DP83856 supports up to an 8k x 16-bit SRAM configuration. This configuration provides a maximum of 16, 32-bit statistics values per port.

| Signal Name                     | Type       | Active | Description                                                                                                               |
|---------------------------------|------------|--------|---------------------------------------------------------------------------------------------------------------------------|
| SA[12:0]                        | O/Z, L     |        | <b>SRAM ADDRESS [12:0]:</b> The SRAM address bus should be directly connected to the fast external SRAM's address inputs. |
| SD[15:0]                        | I/O/Z/P, L |        | <b>SRAM DATA [15:0]:</b> The SRAM data bus, should be directly connected to the fast external SRAM's data pins.           |
| $\text{SR}/\overline{\text{W}}$ | O/Z, L     |        | <b>SRAM READ-WRITE:</b> Should be directly connected to the fast external SRAM's write enable pin. Read = 1, Write = 0.   |
| $\overline{\text{SCS}}$         | O/Z, L     | Low    | <b>SRAM CHIP SELECT:</b> Should be directly connected to the fast external SRAM's active low chip select pin.             |
| $\overline{\text{SOE}}$         | O/Z, L     | Low    | <b>SRAM OUTPUT ENABLE:</b> Should be directly connected to the fast external SRAM's (active low) output enable pin.       |

## 2.0 Pin Description (Continued)

### 2.3 TRANSMIT BUS AND MANAGEMENT BUS

| Signal Name | Type | Active | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|-------------|------|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| TXD[3:0]    | I    |        | <b>TRANSMIT DATA [3:0]:</b> Transfers data from a local DP83850 to the DP83856. TXD[3:0] is synchronous to the local clock signal LC, and is framed by the transmit ready signal TX_RDY.                                                                                                                                                                                                                                                                                                                                                                                                            |
| TX_RDY      | I    | High   | <b>TRANSMIT DATA READY:</b> Asserted by a local DP83850 when non-idle symbols are repeated on any of the DP83850's output ports. The DP83856 uses this signal as a framing signal for transmit data, transmit error, management data, management error, collision, data valid, and as an enable for carrier and network utilization timing.                                                                                                                                                                                                                                                         |
| TX_ER       | I    | High   | <b>TRANSMIT DATA ERROR:</b> Asserted by a local DP83850 when a transmit error occurs. The DP83856 monitors this signal to determine if the current reception was a Symbol Code violation error. TX_ER is synchronous to the local clock signal LC.                                                                                                                                                                                                                                                                                                                                                  |
| IR_COL      | I    | Low    | <b>INTER REPEATER COLLISION:</b> Asserted by any (all) DP83850s in the system which are currently experiencing a collision. The DP83856 monitors this signal during TX_RDY valid, and uses the information in statistics processing and collision counting.                                                                                                                                                                                                                                                                                                                                         |
| IRD_V       | I    | Low    | <b>INTER REPEATER DATA VALID:</b> Asserted by any DP83850 in the system which has won the Inter Repeater Bus arbitration and is transmitting valid data symbols. The DP83856 monitors this line at the beginning of the frame to establish whether the frame is a false carrier event. If TX_RDY is valid and IRDV is invalid when the DP83856 samples the IRDV line, then a false carrier event is counted.                                                                                                                                                                                        |
| MD[3:0]     | I    |        | <b>MANAGEMENT DATA [3:0]:</b> Data which is sourced by any DP83850 in the system that has won the Inter Repeater Bus arbitration. This data is synchronous to the management clock M_CLK, and is framed by the transmit ready signal TX_RDY. The DP83856 uses this data to determine the source of the current data stream (DP83850 RID number and Port number).                                                                                                                                                                                                                                    |
| M_DV        | I    | Low    | <b>MANAGEMENT DATA VALID:</b> Asserted by any DP83850 in the system which has won the Inter Repeater Bus arbitration when it places valid data on MD[3:0]. The DP83856 monitors this line when TX_RDY is valid to determine when to latch the DP83850 RID number and port number for the current reception. M_DV is synchronous to M_CLK.                                                                                                                                                                                                                                                           |
| M_CLK       | I    |        | <b>MANAGEMENT CLOCK:</b> All data transfers on the management bus are synchronized to the rising edge of this clock. M_CLK is a reference 25 MHz clock used to latch the active DP83850, port, and elasticity buffer errors for the current packet reception. M_CLK is sourced by any DP83850 in the system which has won the Inter Repeater Bus arbitration.                                                                                                                                                                                                                                       |
| M_ER        | I    | Low    | <b>MANAGEMENT ERROR:</b> Asserted by any DP83850 in the system which has won the Inter Repeater Bus arbitration when a data rate mismatch error occurs (elasticity buffer over/underrun). The DP83856 monitors this line during TX_RDY valid to determine if the current frame contains a data rate management error. M_ER is synchronous to M_CLK. Note that on data rate mismatch errors, M_ER will not be asserted until after the DP83850 number and port number have been sent to the DP83856 from the DP83850, but will be sent prior to the end of the frame (before TX_RDY is de-asserted). |

### 2.4 MII INTERFACE

| Signal Name | Type     | Active | Description                                                                                                                                                                                                                                                                                         |
|-------------|----------|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| RDC         | O/Z, L   |        | <b>REGISTER DATA CLOCK:</b> A 2.5 MHz clock which is continuously output from the DP83856. Used to synchronize data transfers on the serial MII register bus.                                                                                                                                       |
| RDIO        | I/O/Z, L |        | <b>REGISTER DATA I/O:</b> Serial MII register data signal. Used to transfer data to and from the DP83856 on MII register accesses. This signal should be buffered onto the backplane, using the RRDIR signal as a direction control for the buffer. The buffer does not require a tri-state enable. |
| RRDIR       | O/Z, L   | High   | <b>RIB REGISTER DIRECTION:</b> Serial MII Register Direction pin to drive an external buffer. The buffer should default to READ, and toggle to WRITE only when the DP83856 is initiating an MII register access.<br>0 = MII Slave (DP83850 or PHY) drives RDIO<br>1 = DP83856 drives RDIO           |
| SDV         | O/Z, L   | Low    | <b>SERIAL DATA VALID:</b> Indicates that a valid MII access is in progress. It is asserted one half clock prior to the start of the cycle and remains valid for one half clock after the cycle is complete.                                                                                         |

## 2.0 Pin Description (Continued)

### 2.5 TEST INTERFACE

| Signal Name             | Type            | Active          | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                         |                 |                 |   |   |   |   |   |   |
|-------------------------|-----------------|-----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------|-----------------|-----------------|---|---|---|---|---|---|
| $\overline{TSTATE}$     | I/P             | Low             | <b>TRI-STATE®:</b> Pulling this pin low puts the DP83856 into a test mode that tri-states all outputs except $\overline{NAND\_E}$ and $\overline{NAND\_O}$ . This allows an external tester to drive all the outputs of the DP83856.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |                         |                 |                 |   |   |   |   |   |   |
| $\overline{TEST\_EN}$   | I/P             | Low             | <b>TEST MODE ENABLE HIGH/LOW OUTPUT TEST:</b> Forces the DP83856's outputs to the High or Low state as defined by the $\overline{TEST\_H\_L}$ pin. This allows automatic test machines to check for outputs stuck at High or Low.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |                         |                 |                 |   |   |   |   |   |   |
| $\overline{TEST\_H\_L}$ | I/P             | Low             | <p><b>TEST MODE OUTPUT HIGH/LOW:</b> When <math>\overline{TEST\_EN}</math> is taken Low, the DP83856's output pins (in two groups) are forced into the High or Low state as defined below:</p> <table border="1" style="margin-left: auto; margin-right: auto;"> <thead> <tr> <th><math>\overline{TEST\_H\_L}</math></th> <th>Group 1 Outputs</th> <th>Group 2 Outputs</th> </tr> </thead> <tbody> <tr> <td>0</td> <td>1</td> <td>0</td> </tr> <tr> <td>1</td> <td>0</td> <td>1</td> </tr> </tbody> </table> <p><u>Group 1 output pin numbers are:</u> 21, 23, 27, 31, 35, 37, 39, 43, 61, 64, 68, 70, 72, 74, 78, 80, 82, 87, 89, 90, 92, 96, 98, 100, 104, 108, 113, 115.</p> <p><u>Group 2 output pin numbers are:</u> 20, 22, 24, 30, 34, 36, 38, 42, 44, 65, 69, 71, 73, 75, 79, 81, 83, 88, 91, 93, 97, 99, 103, 105, 114, 116.</p> | $\overline{TEST\_H\_L}$ | Group 1 Outputs | Group 2 Outputs | 0 | 1 | 0 | 1 | 0 | 1 |
| $\overline{TEST\_H\_L}$ | Group 1 Outputs | Group 2 Outputs |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                         |                 |                 |   |   |   |   |   |   |
| 0                       | 1               | 0               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                         |                 |                 |   |   |   |   |   |   |
| 1                       | 0               | 1               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                         |                 |                 |   |   |   |   |   |   |
| $\overline{NAND\_E}$    | O, L            | Low             | <b>NAND TREE EVEN INPUTS OUTPUT:</b> The logical NAND of all of the even numbered inputs (except the test input $\overline{TEST\_EN}$ ) and $\overline{RST}$ . If all of the inputs are High, the output will go Low. If any of the inputs are Low, the output will remain High.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                         |                 |                 |   |   |   |   |   |   |
| $\overline{NAND\_O}$    | O, L            | Low             | <b>NAND TREE ODD INPUTS OUTPUT:</b> The logical NAND of all of the odd numbered inputs (except the test inputs $\overline{TSTATE}$ , $\overline{TEST\_H\_L}$ and LC). If all of the inputs are High, the output will go Low. If any of the inputs are Low, the output will remain High.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |                         |                 |                 |   |   |   |   |   |   |

### 2.6 MISCELLANEOUS PINS

| Signal Name      | Type | Active | Description                                                                                                                                                                                                                                                                                                                              |
|------------------|------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| LC               | I    |        | <b>LOCAL CLOCK:</b> Primary clock for DP83856 device. All DP83856 internal state machines run off LC. This clock must be the same local clock used to drive the local DP83850 because the TX signals (to which the DP83856 must be synchronized) are all synchronous to the local clock. Must be a 25.000 MHz, 40/60 duty cycle, 50 ppm. |
| $\overline{RST}$ | I    | Low    | <b>RESET:</b> The DP83856 is reset when this signal is asserted low. Asserting this signal will cause all DP83856 state machines and registers to enter their reset state. The statistics SRAM is not cleared by $\overline{RST}$ , it must be cleared/preset by software.                                                               |
| RES1             | O    |        | <b>RESERVED OUTPUT 1:</b> No Connect.                                                                                                                                                                                                                                                                                                    |
| RES2             | O    |        | <b>RESERVED OUTPUT 2:</b> No Connect.                                                                                                                                                                                                                                                                                                    |

### 2.7 PIN TYPE DESIGNATION

| Type       | Description                                                                                  |
|------------|----------------------------------------------------------------------------------------------|
| I          | Input buffer.                                                                                |
| I/P        | Input buffer with internal pull-up resistor.                                                 |
| O, L       | Output buffer, low drive (4 mA).                                                             |
| O/Z, L     | Output buffer with high impedance capability, low drive (4 mA).                              |
| I/O/Z, L   | Bi-directional buffer with high impedance capability, low drive (4 mA).                      |
| I/O/Z, M   | Bi-directional buffer with high impedance capability, medium drive (12 mA).                  |
| I/O/Z/P, L | Bi-directional buffer with high impedance capability and pull-up resistor, low drive (4 mA). |

## 3.0 Functional Description

The following sections describe the different functional blocks of the DP83856 100 Mb/s Repeater Information Base. Referring to the block diagram on page 2 of this data-sheet, the DP83856 is used in conjunction with a number of DP83850s, a management CPU and a fast (20 ns) 8k x 16 bit SRAM. The DP83856 collects and maintains network management statistics from the connected DP83850s and makes them available to the management CPU.

### 3.1 STATISTICS GENERATION

Inputs to the DP83856 Statistic Generation block include the Inter Repeater Bus signals, Management Bus signals and TX bus signals. These signals provide the data streams necessary to create all the statistics collected by the DP83856. The DP83856 uses the fast external SRAM to hold statistics for the current packet reception. Statistics for the current receive packet are collected as follows:

#### 3.1.1 Octet Derived

The majority of the statistics are a function of the octet count. Statistics based on octet counts imply that a valid SFD has been detected and an accurate count of the number of data bytes in the packet are available.

The DP83856 Statistic Generation module has an SFD detect block, which indicates that a valid SFD has been detected so that the octet counter can be enabled. The Source Address latch is used to store the source address of the current packet, so that a comparison to the previous source address can be made at the end of the packet reception. Octet derived statistics include:

- Frames
- Octets
- FCS Errors
- Alignment Errors
- Frames Too Long
- Source Address
- Source Address Changes

#### 3.1.2 Carrier Derived

Other statistics are a function of carrier. Carrier derived statistics have a high probability of occurring on activity bursts which do not include a valid SFD. To ensure accurate statistic gathering a carrier based detection scheme is implemented. A nibble counter is used to calculate the length of the carrier, which is used to create the carrier derived statistics.

The DP83856 employs 32-bit counters for network utilization, false carrier events, and collisions. All of these counters monitor events for all ports, i.e. they are an aggregate of the total repeater events.

Carrier derived statistics gathered by the DP83856 include:

- Runts
- Very Long Events (jabber)
- Network Utilization
- Repeater False Carrier Events
- Repeater Collisions (per port collision map obtained from DP83850s)

#### 3.1.3 DP83850 Notified

For a few of the required statistics the DP83856 has no way of determining the occurrence of that event. These statistics are obtained by notification from the connected DP83850s. DP83850 notified statistics include:

- Data Rate Mismatches
- Symbol Code Violations

#### 3.1.4 Collision Counter

The DP83856 has a 32-bit counter which is incremented any time the repeater experiences a collision. This counter is used to keep track of total number of collisions happening on the repeater.

#### 3.1.5 Network Utilization Counter

The Network Utilization counter is a 32-bit counter that counts nibbles when TX\_RDY is active. The network utilization counter will count all nibbles relative to any packet activity. This includes any short events, runts, and even noise on the segment. The network utilization counter is used to measure the packet activity relative to the overall network bandwidth. Since the nibbles occurs at a 25 MHz rate, during a one second update there could be up to a maximum of 25 million nibbles that could be recorded.

As an example, software can do a one second read update on the 32-bit network utilization counter, and read a 10 million value count. The network utilization can be calculated as  $10 \text{ mil.} / 25 \text{ mil. (max. total)} = 0.4$  or 40 per cent of the network bandwidth is utilized on that segment.

#### 3.1.6 False Carrier Counter

The DP83856 has a 32-bit counter which is incremented any time the repeater experiences a false carrier event. This counter is used to keep track of the total number of false carrier events occurring on the segment.

### 3.2 SRAM INTERFACE

The SRAM interface provides the logic required to communicate with the fast external SRAM.

The interface between the DP83856 and the fast external SRAM is very straightforward. The fast external SRAM is a dedicated block of memory directly accessed only by the DP83856. The DP83856 provides the address capability for 8k x 16 bits of SRAM. In this configuration the DP83856 can store up to sixteen 32-bit statistics per port.

Figure 1 shows a memory map for the 8k x 16 configuration. For each port there are 11 statistics defined which are stored in SRAM. Ten of these statistics are 32-bit values, and one is a 48-bit value (Last Source Address).

Last Source Address is stored as two 32-bit values for simplicity of hardware implementation. All statistics are stored in big endian mode.

The DP83856 can be directly connected to the SRAM; there is no need for buffering between the DP83856 and the SRAM. The DP83856 requires fast SRAM with a maximum access time of 20 ns.

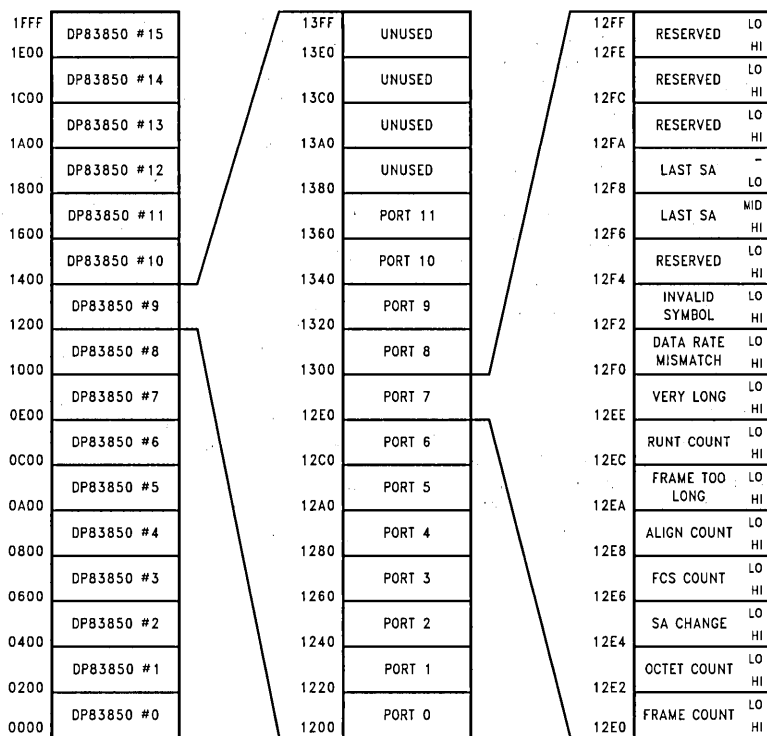
The SRAM interface block contains the address and data multiplexers to select between CPU and Statistic Update accesses. Data is multiplexed under control of the SRAM arbiter.

### 3.3 SRAM ARBITER

The SRAM arbiter controls the SRAM data multiplexers depending on what type of access is being performed and creates all of the control signals for the SRAM, ensuring the timing is correct. There are three events that result in SRAM arbitration:

- End of packet request\_Runt (Statistic Update State Machine)
- End of packet request\_Legal Length or greater (Statistic Update State Machine)
- CPU request (read or write)

### 3.0 Functional Description (Continued)



TL/F/12392-4

FIGURE 1. Memory Map for the DP83856 Statistics SRAM

The arbiter assigns highest priority to EOP-runs and lowest priority to CPU requests. For single statistic reads, the arbiter produces two 16-bit locked read cycles on the SRAM to form the 32-bit value. For block reads the SRAM arbiter re-arbitrates after each 32-bit SRAM read (two 16-bit locked reads) to allow any higher priority event access to the SRAM. Writes to the SRAM must always be word (16-bit) accesses: byte writes are not supported.

#### 3.4 INTERRUPT GENERATION AND CONTROL

There are four events that can generate an interrupt:

- SRAM access complete
- MII register access complete
- Invalid MII register read
- DP83856 error

The DP83856 provides one interrupt line ( $\overline{\text{CINT}}$ ) that is shared for all interrupts. The interrupt is an active low, level sensitive signal. Interrupts are generated based on a valid event occurring with the appropriate mask bit set and global interrupt bit set. Interrupts are cleared by reading the interrupt register.

The "SRAM access complete" interrupt signifies that the current SRAM request has been serviced and the data is in the SRAM Read Data Register(s) (10h–40h).

The "MII register access complete" interrupt signifies that the current MII register request has been serviced and the data is in the MII Read Data Register(s) (A0h–ACh).

The "Invalid MII register read" interrupt is generated based on the DP83856 detecting an error while performing an MII read access. Per the MII register protocol in IEEE 802.3u/D5, Clause 22, the DP83856 looks for a leading 0 on reads; if it does not see a leading 0, it flags the read as invalid and generates the interrupt.

The "DP83856 error" interrupt signifies that the device has dropped management data due to invalid inter-frame gap (IFG) spacing between packets. The minimum IFG specified in the IEEE 802.3u/D5, Clause 21A.3 is 0.96  $\mu\text{s}$ . The DP83856 may drop management data if the IFG drops below 0.64  $\mu\text{s}$ . This event is not expected to happen on well formed 100 Mb/s networks.

#### 3.5 MII REGISTER INTERFACE

The MII register interface block is a state machine that performs accesses to DP83850 and Physical Layer Device registers (read and write) based on requests from the CPU.

This interface uses the IEEE 802.3u clause 22 MII compliant serial interface protocol.

The MII Register Interface eliminates the need for the CPU to talk directly to the DP83850 and Physical Layer Device registers. The amount of spare management CPU processing bandwidth is therefore increased.

### 3.0 Functional Description (Continued)

The CPU provides the opcode, type of access (read or write), register address, and device ID to the MII Interface Register, and then asserts a start command by writing a 1 to bit 0 (MII\_ACC) of the Configuration Register.

MI1 protocol for performing reads and writes are as follows:  
READ

```
<01> <10> <AAAA> <RRRRR> <line turn-around>
<0> <xxxx xxxx xxxx xxx>
```

where <01> is a start bit sequence, <10> is a read opcode, <AAAA> is the device address (up to 32 devices), <RRRRR> is the register address (up to 32 registers), <line turn-around> is a clock cycle allowed for turn-around of the data bus, <0> is a leading 0, and <xx ... xx> is 16 bits of data.

WRITE

```
<01> <01> <AAAA> <RRRRR>
<10> <xxxx xxxx xxxx xxx>
```

where <01> is a start bit sequence, <01> is a write opcode, <AAAA> is the device address (up to 32 devices), <RRRRR> is the register address (up to 32 registers), <10> is a leading 10, and <xx ... xx> is 16 bits of data.

Refer to the IEEE 802.3u D5 draft document for more details on the MII interface, its function and timing.

### 3.6 CPU REGISTER BLOCK

The CPU register block provides the system management CPU access to all of the data in the DP83856, SRAM, connected DP83850s and Physical Layer Devices.

### 3.7 MANAGEMENT DATA INTERFACE

For every network event, the DP83850 device with port N will send information required by the DP83856 to perform statistics gathering.

The information required by the DP83856 is transferred on the Management Data Interface, which consists of a nibble wide data bus with synchronous clock (MD[3:0] and M\_CK), plus a framing signal (M\_DV) and an error flag (M\_ER).

Data is transferred in 3 nibbles (per network event). The first data nibble contains the lowest significant 4 bits of the DP83850 RID Number, the second contains the most significant bit of the DP83850 RID number and the third contains the number of the receiving port (port N). The M\_ER signal indicates elasticity buffer errors (due to under-run or over-run) in the DP83850.

## 4.0 Registers

All the DP83856 registers are directly addressable by the system management CPU. Although some bits in the Configuration Register have been allocated to a register paging scheme, these are not currently used (available for future expansion) and should always be set to zero.

All register accesses are word (16-bit) wide: byte access is not supported. The addresses given in the tables below assume that the user has connected the DP83856 to a man-

agement CPU in the normal 16-bit manner with address bits A1–A7 from the CPU connected to bits CA1–CA7 on the DP83856. The addresses are thus the offset from the base address at which the DP83856 is located in the system.

To be consistent with normal address bus labeling practice, since the DP83956 only supports 16-bit accesses, no pin A0 is supplied.

### 4.1 REGISTER MEMORY MAP

| Address | Register                              | Access |
|---------|---------------------------------------|--------|
| 00h     | Configuration Register                | R/W    |
| 02h     | Interrupt                             | R/W    |
| 04h     | Reserved                              |        |
| 06h     | SRAM Interface                        | R/W    |
| 08h     | MII Management Interface              | R/W    |
| 0Ah     | SRAM Write Data                       | R/W    |
| 0Ch     | MII Write Data                        | R/W    |
| 0Eh     | Device ID                             | R only |
| 10h     | Frame Count High Read                 | R/W    |
| 12h     | Frame Count Low Read                  | R/W    |
| 14h     | Octet Count High Read                 | R/W    |
| 16h     | Octet Count Low Read                  | R/W    |
| 18h     | Source Address Change Count High Read | R/W    |
| 1Ah     | Source Address Change Count Low Read  | R/W    |
| 1Ch     | FCS Error Count High Read             | R/W    |
| 1Eh     | FCS Error Count Low Read              | R/W    |
| 20h     | Alignment Error Count High Read       | R/W    |
| 22h     | Alignment Error Count Low Read        | R/W    |
| 24h     | Frame Too Long Count High Read        | R/W    |
| 26h     | Frame Too Long Count Low Read         | R/W    |
| 28h     | Runt Count High Read                  | R/W    |
| 2Ah     | Runt Count Low Read                   | R/W    |
| 2Ch     | Very Long Event Count High Read       | R/W    |
| 2Eh     | Very Long Event Count Low Read        | R/W    |
| 30h     | Data Rate Mismatch Count High Read    | R/W    |
| 32h     | Data Rate Mismatch Count Low Read     | R/W    |
| 34h     | Invalid Symbol Count High Read        | R/W    |
| 36h     | Invalid Symbol Count Low Read         | R/W    |
| 38h     | Reserved                              |        |
| 3Ah     | Reserved                              |        |
| 3Ch     | Source Address High Read              | R/W    |
| 3Eh     | Source Address Mid Read               | R/W    |
| 40h     | Source Address Low Read               | R/W    |
| 42h     | Reserved                              |        |
| 44h–7Eh | Reserved                              |        |

## 4.0 Registers (Continued)

### 4.1 REGISTER MEMORY MAP (Continued)

| Address  | Register                                                   | Access |
|----------|------------------------------------------------------------|--------|
| 80h      | Carrier Count Register                                     | R/W    |
| 82h      | Oct_Nib Count Register                                     | R/W    |
| 84h–8Eh  | Reserved                                                   |        |
| 90h      | Repeater Collisions High Read                              | R/W    |
| 92h      | Repeater Collisions Low Read                               | R/W    |
| 94h      | Network Utilization High Read                              | R/W    |
| 96h      | Network Utilization Low Read                               | R/W    |
| 98h      | False Carrier High Read                                    | R/W    |
| 9Ah      | False Carrier Low Read                                     | R/W    |
| 9Ch–9Eh  | Reserved                                                   |        |
| A0h      | MII Read Data / Port 0-11 Short Event High Block Read Data | R only |
| A2h      | DP83850 Port 0-11 Short Event Low Block Read Data          | R only |
| A4h      | DP83850 Port 0-11 Late Event High Block Read Data          | R only |
| A6h      | DP83850 Port 0-11 Late Event Low Block Read Data           | R only |
| A8h      | DP83850 Port 0-11 Collision High Block Read Data           | R only |
| AAh      | DP83850 Port 0-11 Collision Low Block Read Data            | R only |
| ACh      | DP83850 Port 0-11 Auto-Partitions Block Read Data          | R only |
| AEh      | Reserved                                                   |        |
| B0h–1FEh | Reserved                                                   |        |

### 4.2 CONFIGURATION REGISTER

Address: 00h

Reset: All bits cleared to zero.

| Bit     | Bit Name    | Access | Description                                                                                                                                                                                                                                                   |
|---------|-------------|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0      | MII_ACC     | R/W    | 0: DP83856 writes 0 after register access completes<br>1: CPU initiates register access by writing 1<br><br>This bit indicates when the current DP83850 or Physical Layer device register access is complete.                                                 |
| D1      | SR_ACC      | R/W    | 0: DP83856 writes 0 after SRAM access completes<br>1: CPU initiates SRAM access by writing 1<br><br>This bit indicates when the current SRAM access is complete.                                                                                              |
| D2      | SR_ACC_TYPE | R/W    | 0: Perform Single Access<br>1: Perform Block Access (Reads Only) All SRAM based statistics will be loaded into SRAM (CPU Addr. 10h–40h)                                                                                                                       |
| D3      | MEN         | R/W    | 0: Statistics gathering disabled<br>1: Statistics gathering enabled<br><br>This bit enables management statistics gathering. Note that subsequent read accesses should use read-modify-write instructions so that the MEN bit is left undisturbed.            |
| D(4:7)  | PAGE_SEL    | R/W    | These bits define which page of the register map the CPU is pointing to. Allows for 16 pages x 256 word registers.<br><br>Always write 0 for compatibility with later versions of DP83856.<br><br>Note: The Page bits are not implemented in current version. |
| D(8:15) | Reserved    | R/W    | Write: 0<br>Read: Undefined                                                                                                                                                                                                                                   |



## 4.0 Registers (Continued)

### 4.3 INTERRUPT REGISTER

Address: 02h

Reset: All bits cleared to zero.

**Note:** Mask bits must be set to enable valid status on corresponding status bits.

| Bit     | Bit Name       | Access | Description                                                                                                                                                                        |
|---------|----------------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D0      | MII_INT_STS    | R      | 0: MII access complete Interrupt not asserted.<br>1: MII access complete Interrupt asserted.<br>Cleared by read of register. Writes ignored.                                       |
| D1      | SR_ACC_STS     | R      | 0: SRAM access complete Interrupt not asserted.<br>1: SRAM access complete Interrupt asserted.<br>Cleared by read of register. Writes ignored.                                     |
| D2      | RIBERR_STS     | R      | 0: DP83856 error Interrupt not asserted.<br>0: MII access complete Interrupt not asserted.<br>1: DP83856 error Interrupt asserted.<br>Cleared by read of register. Writes ignored. |
| D3      | MII_RD_ERR_STS | R      | 0: MII register read error Interrupt not asserted.<br>1: MII register read error Interrupt asserted.<br>Cleared by read of register. Writes ignored.                               |
| D4      | MII_INT_MSK    | R/W    | 0: Mask MII access complete Interrupt.<br>1: Enable MII access complete Interrupt.                                                                                                 |
| D5      | SR_ACC_MSK     | R/W    | 0: Mask SRAM access complete Interrupt.<br>1: Enable SRAM access complete Interrupt.                                                                                               |
| D6      | RIBERR_MSK     | R/W    | 0: Mask DP83856 error Interrupt.<br>1: Enable DP83856 error Interrupt.                                                                                                             |
| D7      | MII_RD_ERR_MSK | R/W    | 0: Mask MII Register Error Interrupt.<br>1: Enable MII Register Error Interrupt.                                                                                                   |
| D8      | INT_EN         | R/W    | 0: Disable $\overline{CINT}$ signal.<br>1: Enable $\overline{CINT}$ signal.<br>This bit is a global enable for the $\overline{CINT}$ signal. It has NO effect on the status bits.  |
| D(9:15) | Reserved       | R/W    | Write: 0<br>Read: Undefined.                                                                                                                                                       |

## 4.0 Registers (Continued)

### 4.4 SRAM INTERFACE REGISTER

Address: 06h

Reset: All bits cleared to zero.

| Bit      | Bit Name   | Access | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|----------|------------|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D(0:4)   | STAT__ACC# | R/W    | These bits set which STATISTIC the SRAM access is destined for. Values are:<br><b>00h:</b> Frame Count<br><b>02h:</b> Octet Count<br><b>04h:</b> SA Change Count<br><b>06h:</b> FCS Error Count<br><b>08h:</b> Alignment Error Count<br><b>0Ah:</b> Frame Too Long Count<br><b>0Ch:</b> Runt Count<br><b>0Eh:</b> Very Long Event Count<br><b>10h:</b> Data Rate Mismatch Count<br><b>12h:</b> Invalid Symbol Count<br><b>14h:</b> Reserved<br><b>16h:</b> Source Address High<br><b>18h:</b> Source Address Low<br><b>1Ah–1Eh:</b> Reserved |
| D(5:6)   | Reserved   | R/W    | Always Write 0                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| D7       | R/W__SRAM  | R/W    | 0: SRAM Write<br>1: SRAM Read<br>This bit defines whether the current CPU SRAM access is a read or a write.                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| D(8:11)  | PORT__ACC# | R/W    | These bits set which PORT number the access is destined for.<br>Valid values are 0h–Bh (12 ports)                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| D(12:15) | RIC__ACC#  | R/W    | These bits set which DP83850 the access is destined for.<br>Valid values are 0h–Fh (16, DP83850s)                                                                                                                                                                                                                                                                                                                                                                                                                                            |

**Note:** This register should NOT be accessed while an SRAM access is in progress (If bit D1 of Configuration Register is 1, then do not access this register).

### 4.5 MII MANAGEMENT INTERFACE REGISTER

Address: 08h

Reset: All bits cleared to zero.

| Bit      | Bit Name      | Access | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------|---------------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D(0:4)   | REG__ADDR     | R/W    | These bits set which register the access is destined for.                                                                                                                                                                                                                                                                                                                                                                                                                          |
| D(5:9)   | DEV__ID       | R/W    | These bits set which DEVICE__ID the access is destined for.                                                                                                                                                                                                                                                                                                                                                                                                                        |
| D(10:11) | OPCODE        | R/W    | <b>OPCODE VALUE:</b> Corresponds to the opcodes defined in the MII specification.<br><b>01:</b> Extended Addressed Mode Write, 16-bit payload.<br><b>10:</b> Extended Addressed Mode Read, 16-bit payload.                                                                                                                                                                                                                                                                         |
| D12      | MII__ACC__TYP | R/W    | <b>MII ACCESS TYPE:</b> Sets the access type to single or block read.<br><b>0:</b> Perform Single Access (All Physical Layer device accesses and all DP83850 accesses except DP83850 counters).<br><b>1:</b> Perform Block Read (DP83850 reads only). All DP83850 based counters will be loaded into registers (Address A0h–ACh). The OPCODE field is 10 for block reads. REG__ADDR is set to register address corresponding to the Port__ShortEvent Counter for the desired port. |
| D(13:15) | Reserved      | R/W    | Write: 0.<br>Read: Undefined.                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

## 4.0 Registers (Continued)

### 4.6 SRAM WRITE DATA REGISTER

Address: 0Ah

Reset: All bits cleared to zero.

| Bit     | Bit Name | Access | Description                                                                                                                                |
|---------|----------|--------|--------------------------------------------------------------------------------------------------------------------------------------------|
| D(0:15) | WR_DATA  | R/W    | This register contains the data to be written on an SRAM write access. SRAM writes should only be performed during DP83856 initialization. |

### 4.7 MII WRITE DATA REGISTER

Address: 0Ch

Reset: All bits cleared to zero.

| Bit     | Bit Name | Access | Description                                                                    |
|---------|----------|--------|--------------------------------------------------------------------------------|
| D(0:15) | WR_DATA  | R/W    | This register contains the data to be written on an MII register write access. |

### 4.8 DEVICE ID REGISTER

Address: 0Eh

Reset: All bits cleared to zero.

| Bit     | Bit Name  | Access | Description                                                                                                               |
|---------|-----------|--------|---------------------------------------------------------------------------------------------------------------------------|
| D(0:3)  | REV_LEVEL | R      | These bits are the Revision level of the device and are embedded into the DP83856 silicon. Reads 0h for initial revision. |
| D(4:7)  | DEVICE ID | R      | These bits are a vendor specific code embedded in the DP83856. Reads 0h for initial revision.                             |
| D(8:15) | Reserved  | R/W    | Write: 0<br>Read: Undefined.                                                                                              |

### 4.9 SRAM READ DATA REGISTERS

Addresses: 10h–40h

Reset: All bits cleared to zero.

| Bit     | Bit Name       | Access | Description                                                |
|---------|----------------|--------|------------------------------------------------------------|
| D(0:15) | SRAM Read Data | R      | Contains data corresponding to the SRAM location selected. |

## 4.0 Registers (Continued)

### 4.10 CARRIER COUNT REGISTER

Address: 80h

Reset: All bits cleared to zero.

| Bit      | Bit Name      | Access | Description                                                                                                                                                       |
|----------|---------------|--------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D(0:13)  | Carrier Count | R/W    | Contains data which is used to preset the carrier counter FOR TEST PURPOSES ONLY. This register can only be written when the MEN bit in the CONFIG register is 0. |
| D(14:15) | Unused        | R/W    | Write: 0<br>Read: Undefined.                                                                                                                                      |

### 4.11 OCT\_NIB COUNT REGISTER

Address: 82h

Reset: All bits cleared to zero.

| Bit      | Bit Name      | Access | Description                                                                                                                                                            |
|----------|---------------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D(0:11)  | Oct_Nib Count | R/W    | Contains data which is used to preset the Octet-Nibble counter FOR TEST PURPOSES ONLY. This register can only be written when the MEN bit in the CONFIG register is 0. |
| D(12:15) | Unused        | R/W    | Write: 0<br>Read: Undefined.                                                                                                                                           |

### 4.12 NETWORK COUNTERS

Addresses: 90h–9Ah

Reset: All bits cleared to zero.

| Bit     | Bit Name     | Access | Description                                                                                                                                                                     |
|---------|--------------|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| D(0:15) | Counter Data | R/W    | Contains data corresponding to the selected counter. Disable the Management function by writing 0 to the MEN, bit D3 in the Config register prior to writing to these counters. |

### 4.13 MII READ DATA REGISTERS

Addresses: A0h–ACh

Reset: Bit value at reset is indeterminate. Will probably read FFh due to bus pull-up.

| Bit     | Bit Name | Access | Description                                                    |
|---------|----------|--------|----------------------------------------------------------------|
| D(15:0) | MII Data | R      | Contains read data corresponding to the MII register selected. |

For single Physical Layer Management register read accesses and single statistic read accesses to connected DP83850s, the read data appears in data register address A0h. When the DP83856 is instructed to do a block statistics read from a connected DP83850, the block of 7 read values is placed in the registers A0h–ACh. The register designations are given in the memory map in Section 4.1.

## 5.0 DP83856 Initialization

### 5.1 SRAM TEST AND INITIALIZATION

System vendors will often desire to test the SRAM as part of power-up initialization prior to enabling management. SRAM accesses are performed as follows:

#### SRAM Writes:

1. Program the SRAM Interface Register (06h) with the desired information.
2. Program the SRAM Write Data Register (0Ah) with the desired data.
3. Program the Configuration Register (00h) to disable management (D3 = 0), perform a single SRAM access (D2 = 0), and initiate the cycle (D1 = 1). (Note block access to SRAM is not allowed for Writes.)
4. Poll the Configuration Register (00h) for SRAM access complete (D1 = 0).
5. Repeat for next write.

#### SRAM Reads:

1. Program the SRAM Interface Register (06h) with the desired information.
2. Program the Configuration Register (00h) to enable management (D3 = 1), perform a single SRAM access (D2 = 0), and initiate the cycle (D1 = 1).
3. Poll the Configuration Register (00h) for SRAM access complete (D1 = 0).
4. Read the appropriate SRAM Read Data Register (10b–40h) to obtain data.
5. Repeat for next read.

Note that there are “holes” in the SRAM space, i.e., locations which are not accessible due to the statistic update implementation. The DP83856 implementation uses 8k x 16 of SRAM. Assuming the base address of the SRAM is 0000h, then the locations (word addresses) which are inaccessible are given in Table I below.

**TABLE I. Unaccessible SRAM Locations  
(8k x 16 Map, word addressing, in hex)**

| DP83850 # | Offset xx4 | Offset xxA | Offset xxC | Offset xxE |
|-----------|------------|------------|------------|------------|
| 0         | 0014       | 001A       | 001C       | 001E       |
| 0         | 0034       | 003A       | 003C       | 003E       |
| 0         | 0054       | 005A       | 005C       | 005E       |
| 0         | 0074       | 007A       | 007C       | 007E       |
| 0         | 0094       | 009A       | 009C       | 009E       |
| 0         | 00B4       | 00BA       | 00BC       | 00BE       |
| 0         | 00D4       | 00DA       | 00DC       | 00DE       |
| 0         | 00F4       | 00FA       | 00FC       | 00FE       |
| 0         | 0114       | 011A       | 011C       | 011E       |
| 0         | 0134       | 013A       | 013C       | 013E       |
| 0         | 0154       | 015A       | 015C       | 015E       |
| 0         | 0174       | 017A       | 017C       | 017E       |
| 0         | 0194       | 019A       | 019C       | 019E       |
| 0         | 01B4       | 01BA       | 01BC       | 01BE       |
| 0         | 01D4       | 01DA       | 01DC       | 01DE       |
| 0         | 01F4       | 01FA       | 01FC       | 01FE       |
| 1         | 0214       | 021A       | 021C       | 021E       |
| 1         | 0234       | 023A       | 023C       | 023E       |
| 1         | 0254       | 025A       | 025C       | 025E       |
| 1         | 0274       | 027A       | 027C       | 027E       |
| 1         | 0294       | 029A       | 029C       | 029E       |
| 1         | 02B4       | 02BA       | 02BC       | 02BE       |
| 1         | 02D4       | 02DA       | 02DC       | 02DE       |
| 1         | 02F4       | 02FA       | 02FC       | 02FE       |
| 1         | 0314       | 031A       | 031C       | 031E       |
| 1         | 0334       | 033A       | 033C       | 033E       |
| 1         | 0354       | 035A       | 035C       | 035E       |
| 1         | 0374       | 037A       | 037C       | 037E       |
| 1         | 0394       | 039A       | 039C       | 039E       |
| 1         | 03B4       | 03BA       | 03BC       | 03BE       |
| 1         | 03D4       | 03DA       | 03DC       | 03DE       |
| 1         | 03F4       | 03FA       | 03FC       | 03FE       |

## 5.0 DP83856 Initialization (Continued)

TABLE I. Unaccessible SRAM Locations (8k x 16 Map, word addressing, in hex) (Continued)

| DP83850 # | Offset xx4 | Offset xxA | Offset xxC | Offset xxE |
|-----------|------------|------------|------------|------------|
| 2         | 0414       | 041A       | 041C       | 041E       |
| 2         | 0434       | 043A       | 043C       | 043E       |
| 2         | 0454       | 045A       | 045C       | 045E       |
| 2         | 0474       | 047A       | 047C       | 047E       |
| 2         | 0494       | 049A       | 049C       | 049E       |
| 2         | 04B4       | 04BA       | 04BC       | 04BE       |
| 2         | 04D4       | 04DA       | 04DC       | 04DE       |
| 2         | 04F4       | 04FA       | 04FC       | 04FE       |
| 2         | 0514       | 051A       | 051C       | 051E       |
| 2         | 0534       | 053A       | 053C       | 053E       |
| 2         | 0554       | 055A       | 055C       | 055E       |
| 2         | 0574       | 057A       | 057C       | 057E       |
| 2         | 0594       | 059A       | 059C       | 059E       |
| 2         | 05B4       | 05BA       | 05BC       | 05BE       |
| 2         | 05D4       | 05DA       | 05DC       | 05DE       |
| 2         | 05F4       | 05FA       | 05FC       | 05FE       |
| 3         | 0614       | 061A       | 061C       | 061E       |
| 3         | 0634       | 063A       | 063C       | 063E       |
| 3         | 0654       | 065A       | 065C       | 065E       |
| 3         | 0674       | 067A       | 067C       | 067E       |
| 3         | 0694       | 069A       | 069C       | 069E       |
| 3         | 06B4       | 06BA       | 06BC       | 06BE       |
| 3         | 06D4       | 06DA       | 06DC       | 06DE       |
| 3         | 06F4       | 06FA       | 06FC       | 06FE       |
| 3         | 0714       | 071A       | 071C       | 071E       |
| 3         | 0734       | 073A       | 073C       | 073E       |
| 3         | 0754       | 075A       | 075C       | 075E       |
| 3         | 0774       | 077A       | 077C       | 077E       |
| 3         | 0794       | 079A       | 079C       | 079E       |
| 3         | 07B4       | 07BA       | 07BC       | 07BE       |
| 3         | 07D4       | 07DA       | 07DC       | 07DE       |
| 3         | 07F4       | 07FA       | 07FC       | 07FE       |

| DP83850 # | Offset xx4 | Offset xxA | Offset xxC | Offset xxE |
|-----------|------------|------------|------------|------------|
| 4         | 0814       | 081A       | 081C       | 081E       |
| 4         | 0834       | 083A       | 083C       | 083E       |
| 4         | 0854       | 085A       | 085C       | 085E       |
| 4         | 0874       | 087A       | 087C       | 087E       |
| 4         | 0894       | 089A       | 089C       | 089E       |
| 4         | 08B4       | 08BA       | 08BC       | 08BE       |
| 4         | 08D4       | 08DA       | 08DC       | 08DE       |
| 4         | 08F4       | 08FA       | 08FC       | 08FE       |
| 4         | 0914       | 091A       | 091C       | 091E       |
| 4         | 0934       | 093A       | 093C       | 093E       |
| 4         | 0954       | 095A       | 095C       | 095E       |
| 4         | 0974       | 097A       | 097C       | 097E       |
| 4         | 0994       | 099A       | 099C       | 099E       |
| 4         | 09B4       | 09BA       | 09BC       | 09BE       |
| 4         | 09D4       | 09DA       | 09DC       | 09DE       |
| 4         | 09F4       | 09FA       | 09FC       | 09FE       |
| 5         | 0A14       | 0A1A       | 0A1C       | 0A1E       |
| 5         | 0A34       | 0A3A       | 0A3C       | 0A3E       |
| 5         | 0A54       | 0A5A       | 0A5C       | 0A5E       |
| 5         | 0A74       | 0A7A       | 0A7C       | 0A7E       |
| 5         | 0A94       | 0A9A       | 0A9C       | 0A9E       |
| 5         | 0AB4       | 0ABA       | 0ABC       | 0ABE       |
| 5         | 0AD4       | 0ADA       | 0ADC       | 0ADE       |
| 5         | 0AF4       | 0AFA       | 0AFC       | 0AFE       |
| 5         | 0B14       | 0B1A       | 0B1C       | 0B1E       |
| 5         | 0B34       | 0B3A       | 0B3C       | 0B3E       |
| 5         | 0B54       | 0B5A       | 0B5C       | 0B5E       |
| 5         | 0B74       | 0B7A       | 0B7C       | 0B7E       |
| 5         | 0B94       | 0B9A       | 0B9C       | 0B9E       |
| 5         | 0BB4       | 0BBA       | 0BBC       | 0BBE       |
| 5         | 0BD4       | 0BDA       | 0BDC       | 0BDE       |
| 5         | 0BF4       | 0BFA       | 0BFC       | 0BFE       |

## 5.0 DP83856 Initialization (Continued)

TABLE I. Unaccessible SRAM Locations (8k x 16 Map, word addressing, in hex) (Continued)

| DP83850 # | Offset xx4 | Offset xxA | Offset xxC | Offset xxE |
|-----------|------------|------------|------------|------------|
| 6         | 0C14       | 0C1A       | 0C1C       | 0C1E       |
| 6         | 0C34       | 0C3A       | 0C3C       | 0C3E       |
| 6         | 0C54       | 0C5A       | 0C5C       | 0C5E       |
| 6         | 0C74       | 0C7A       | 0C7C       | 0C7E       |
| 6         | 0C94       | 0C9A       | 0C9C       | 0C9E       |
| 6         | 0CB4       | 0CBA       | 0CBC       | 0CBE       |
| 6         | 0CD4       | 0CDA       | 0CDC       | 0CDE       |
| 6         | 0CF4       | 0CFA       | 0CFC       | 0CFE       |
| 6         | 0D14       | 0D1A       | 0D1C       | 0D1E       |
| 6         | 0D34       | 0D3A       | 0D3C       | 0D3E       |
| 6         | 0D54       | 0D5A       | 0D5C       | 0D5E       |
| 6         | 0D74       | 0D7A       | 0D7C       | 0D7E       |
| 6         | 0D94       | 0D9A       | 0D9C       | 0D9E       |
| 6         | 0DB4       | 0DBA       | 0DBC       | 0DBE       |
| 6         | 0DD4       | 0DDA       | 0DDC       | 0DDE       |
| 6         | 0DF4       | 0DFA       | 0DFC       | 0DFE       |
| 7         | 0E14       | 0E1A       | 0E1C       | 0E1E       |
| 7         | 0E34       | 0E3A       | 0E3C       | 0E3E       |
| 7         | 0E54       | 0E5A       | 0E5C       | 0E5E       |
| 7         | 0E74       | 0E7A       | 0E7C       | 0E7E       |
| 7         | 0E94       | 0E9A       | 0E9C       | 0E9E       |
| 7         | 0EB4       | 0EBA       | 0EBC       | 0EBE       |
| 7         | 0ED4       | 0EDA       | 0EDC       | 0EDE       |
| 7         | 0EF4       | 0EFA       | 0EFC       | 0EFE       |
| 7         | 0F14       | 0F1A       | 0F1C       | 0F1E       |
| 7         | 0F34       | 0F3A       | 0F3C       | 0F3E       |
| 7         | 0F54       | 0F5A       | 0F5C       | 0F5E       |
| 7         | 0F74       | 0F7A       | 0F7C       | 0F7E       |
| 7         | 0F94       | 0F9A       | 0F9C       | 0F9E       |
| 7         | 0FB4       | 0FBA       | 0FBC       | 0FBE       |
| 7         | 0FD4       | 0FDA       | 0FDC       | 0FDE       |
| 7         | 0FF4       | 0FFA       | 0FFC       | 0FFE       |

| DP83850 # | Offset xx4 | Offset xxA | Offset xxC | Offset xxE |
|-----------|------------|------------|------------|------------|
| 8         | 1014       | 101A       | 101C       | 101E       |
| 8         | 1034       | 103A       | 103C       | 103E       |
| 8         | 1054       | 105A       | 105C       | 105E       |
| 8         | 1074       | 107A       | 107C       | 107E       |
| 8         | 1094       | 109A       | 109C       | 109E       |
| 8         | 10B4       | 10BA       | 10BC       | 10BE       |
| 8         | 10D4       | 10DA       | 10DC       | 10DE       |
| 8         | 10F4       | 10FA       | 10FC       | 10FE       |
| 8         | 1114       | 111A       | 111C       | 111E       |
| 8         | 1134       | 113A       | 113C       | 113E       |
| 8         | 1154       | 115A       | 115C       | 115E       |
| 8         | 1174       | 117A       | 117C       | 117E       |
| 8         | 1194       | 119A       | 119C       | 119E       |
| 8         | 11B4       | 11BA       | 11BC       | 11BE       |
| 8         | 11D4       | 11DA       | 11DC       | 11DE       |
| 8         | 11F4       | 11FA       | 11FC       | 11FE       |
| 9         | 1214       | 121A       | 121C       | 121E       |
| 9         | 1234       | 123A       | 123C       | 123E       |
| 9         | 1254       | 125A       | 125C       | 125E       |
| 9         | 1274       | 127A       | 127C       | 127E       |
| 9         | 1294       | 129A       | 129C       | 129E       |
| 9         | 12B4       | 12BA       | 12BC       | 12BE       |
| 9         | 12D4       | 12DA       | 12DC       | 12DE       |
| 9         | 12F4       | 12FA       | 12FC       | 12FE       |
| 9         | 1314       | 131A       | 131C       | 131E       |
| 9         | 1334       | 133A       | 133C       | 133E       |
| 9         | 1354       | 135A       | 135C       | 135E       |
| 9         | 1374       | 137A       | 137C       | 137E       |
| 9         | 1394       | 139A       | 139C       | 139E       |
| 9         | 13B4       | 13BA       | 13BC       | 13BE       |
| 9         | 13D4       | 13DA       | 13DC       | 13DE       |
| 9         | 13F4       | 13FA       | 13FC       | 13FE       |

## 5.0 DP83856 Initialization (Continued)

TABLE I. Unaccessible SRAM Locations (8k x 16 Map, word addressing, in hex) (Continued)

| DP83850 # | Offset xx4 | Offset xxA | Offset xxC | Offset xxE |
|-----------|------------|------------|------------|------------|
| 10        | 1414       | 141A       | 141C       | 141E       |
| 10        | 1434       | 143A       | 143C       | 143E       |
| 10        | 1454       | 145A       | 145C       | 145E       |
| 10        | 1474       | 147A       | 147C       | 147E       |
| 10        | 1494       | 149A       | 149C       | 149E       |
| 10        | 14B4       | 14BA       | 14BC       | 14BE       |
| 10        | 14D4       | 14DA       | 14DC       | 14DE       |
| 10        | 14F4       | 14FA       | 14FC       | 14FE       |
| 10        | 1514       | 151A       | 151C       | 151E       |
| 10        | 1534       | 153A       | 153C       | 153E       |
| 10        | 1554       | 155A       | 155C       | 155E       |
| 10        | 1574       | 157A       | 157C       | 157E       |
| 10        | 1594       | 159A       | 159C       | 159E       |
| 10        | 15B4       | 15BA       | 15BC       | 15BE       |
| 10        | 15D4       | 15DA       | 15DC       | 15DE       |
| 10        | 15F4       | 15FA       | 15FC       | 15FE       |
| 11        | 1614       | 161A       | 161C       | 161E       |
| 11        | 1634       | 163A       | 163C       | 163E       |
| 11        | 1654       | 165A       | 165C       | 165E       |
| 11        | 1674       | 167A       | 167C       | 167E       |
| 11        | 1694       | 169A       | 169C       | 169E       |
| 11        | 16B4       | 16BA       | 16BC       | 16BE       |
| 11        | 16D4       | 16DA       | 16DC       | 16DE       |
| 11        | 16F4       | 16FA       | 16FC       | 16FE       |
| 11        | 1714       | 171A       | 171C       | 171E       |
| 11        | 1734       | 173A       | 173C       | 173E       |
| 11        | 1754       | 175A       | 175C       | 175E       |
| 11        | 1774       | 177A       | 177C       | 177E       |
| 11        | 1794       | 179A       | 179C       | 179E       |
| 11        | 17B4       | 17BA       | 17BC       | 17BE       |
| 11        | 17D4       | 17DA       | 17DC       | 17DE       |
| 11        | 17F4       | 17FA       | 17FC       | 17FE       |

| DP83850 # | Offset xx4 | Offset xxA | Offset xxC | Offset xxE |
|-----------|------------|------------|------------|------------|
| 12        | 1814       | 181A       | 181C       | 181E       |
| 12        | 1834       | 183A       | 183C       | 183E       |
| 12        | 1854       | 185A       | 185C       | 185E       |
| 12        | 1874       | 187A       | 187C       | 187E       |
| 12        | 1894       | 189A       | 189C       | 189E       |
| 12        | 18B4       | 18BA       | 18BC       | 18BE       |
| 12        | 18D4       | 18DA       | 18DC       | 18DE       |
| 12        | 18F4       | 18FA       | 18FC       | 18FE       |
| 12        | 1914       | 191A       | 191C       | 191E       |
| 12        | 1934       | 193A       | 193C       | 193E       |
| 12        | 1954       | 195A       | 195C       | 195E       |
| 12        | 1974       | 197A       | 197C       | 197E       |
| 12        | 1994       | 199A       | 199C       | 199E       |
| 12        | 19B4       | 19BA       | 19BC       | 19BE       |
| 12        | 19D4       | 19DA       | 19DC       | 19DE       |
| 12        | 19F4       | 19FA       | 19FC       | 19FE       |
| 13        | 1A14       | 1A1A       | 1A1C       | 1A1E       |
| 13        | 1A34       | 1A3A       | 1A3C       | 1A3E       |
| 13        | 1A54       | 1A5A       | 1A5C       | 1A5E       |
| 13        | 1A74       | 1A7A       | 1A7C       | 1A7E       |
| 13        | 1A94       | 1A9A       | 1A9C       | 1A9E       |
| 13        | 1AB4       | 1ABA       | 1ABC       | 1ABE       |
| 13        | 1AD4       | 1ADA       | 1ADC       | 1ADE       |
| 13        | 0AF4       | 0AFA       | 0AFC       | 0AFE       |
| 13        | 1B14       | 1B1A       | 1B1C       | 1B1E       |
| 13        | 1B34       | 1B3A       | 1B3C       | 1B3E       |
| 13        | 1B54       | 1B5A       | 1B5C       | 1B5E       |
| 13        | 1B74       | 1B7A       | 1B7C       | 1B7E       |
| 13        | 1B94       | 1B9A       | 1B9C       | 1B9E       |
| 13        | 1BB4       | 1BBA       | 1BBC       | 1BBE       |
| 13        | 1BD4       | 1BDA       | 1BDC       | 1BDE       |
| 13        | 1BF4       | 1BFA       | 1BFC       | 1BFE       |



## 5.0 DP83856 Initialization (Continued)

**TABLE I. Unaccessible SRAM Locations**  
(8k x 16 Map, word addressing, in hex) (Continued)

| DP83850 # | Offset xx4 | Offset xxA | Offset xxC | Offset xxE |
|-----------|------------|------------|------------|------------|
| 14        | 1C14       | 1C1A       | 1C1C       | 1C1E       |
| 14        | 1C34       | 1C3A       | 1C3C       | 1C3E       |
| 14        | 1C54       | 1C5A       | 1C5C       | 1C5E       |
| 14        | 1C74       | 1C7A       | 1C7C       | 1C7E       |
| 14        | 1C94       | 1C9A       | 1C9C       | 1C9E       |
| 14        | 1CB4       | 1CBA       | 1CBC       | 1CBE       |
| 14        | 1CD4       | 1CDA       | 1CDC       | 1CDE       |
| 14        | 1CF4       | 1CFA       | 1CFC       | 1CFE       |
| 14        | 1D14       | 1D1A       | 1D1C       | 1D1E       |
| 14        | 1D34       | 1D3A       | 1D3C       | 1D3E       |
| 14        | 1D54       | 1D5A       | 1D5C       | 1D5E       |
| 14        | 1D74       | 1D7A       | 1D7C       | 1D7E       |
| 14        | 1D94       | 1D9A       | 1D9C       | 1D9E       |
| 14        | 1DB4       | 1DBA       | 1DBC       | 1DBE       |
| 14        | 1DD4       | 1DDA       | 1DDC       | 1DDE       |
| 14        | 1DF4       | 1DFA       | 1DFC       | 1DFE       |
| 15        | 1E14       | 1E1A       | 1E1C       | 1E1E       |
| 15        | 1E34       | 1E3A       | 1E3C       | 1E3E       |
| 15        | 1E54       | 1E5A       | 1E5C       | 1E5E       |
| 15        | 1E74       | 1E7A       | 1E7C       | 1E7E       |
| 15        | 1E94       | 1E9A       | 1E9C       | 1E9E       |
| 15        | 1EB4       | 1EBA       | 1EBC       | 1EBE       |
| 15        | 1ED4       | 1EDA       | 1EDC       | 1EDE       |
| 15        | 1EF4       | 1EFA       | 1EFC       | 1EFE       |
| 15        | 1F14       | 1F1A       | 1F1C       | 1F1E       |
| 15        | 1F34       | 1F3A       | 1F3C       | 1F3E       |
| 15        | 1F54       | 1F5A       | 1F5C       | 1F5E       |
| 15        | 1F74       | 1F7A       | 1F7C       | 1F7E       |
| 15        | 1F94       | 1F9A       | 1F9C       | 1F9E       |
| 15        | 1FB4       | 1FBA       | 1FBC       | 1FBE       |
| 15        | 1FD4       | 1FDA       | 1FDC       | 1FDE       |
| 15        | 1FF4       | 1FFA       | 1FFC       | 1FFE       |

### 5.2 NETWORK COUNTER INITIALIZATION

The network counters described in Sections 3.1.4 (Collision Counter), 3.1.5 (Network Utilization Counter), and 3.1.6 (False Carrier Counter) are cleared by power-up reset and require no further initialization.

### 5.3 INTERRUPT INITIALIZATION AND USAGE

The DP83856 supports, but does not require, the use of external interrupts. Interrupts are initialized by programming the Interrupt Register (02h). Interrupt generation and control is described in Section 3.4.

The following is one suggestion on how to use interrupts:

First of all, the agent will probably want to receive updated statistic information based on a timer tick, say once every second.

Software will want to get all of the statistics for each port, so it needs to talk to both the DP83856 and the DP83850.

Using the Block SRAM Read feature of the DP83856, software will have to wait on the order of 10  $\mu$ s for the DP83856 to complete the SRAM dump per port. Thus, it does not seem worthwhile to use interrupts given the overhead of getting in/out of the interrupt service routine.

Using the Block MII Read feature of the DP83856, software will have to wait on the order of 100  $\mu$ s for the DP83856 to complete the DP83850 register dump. (7 MII accesses @ 32 x 400 ns per access, plus overhead) thus, this is where interrupt usage is recommended. Allow software go off and do something else while the DP83850 registers are being read.

The next issue is the MII read error. Software will probably poll the stack of hubs periodically, looking for the addition or removal of a hub from the stack. It is assumed that this polling routine is not executed very often, perhaps once every 5s or 10s. MII\_RD\_ERR interrupts should only be generated by accessing hubs that are not there (although damaged units may also not respond). The point is that an insignificant number of MII\_RD\_ERR interrupts should be generated, and thus very little bandwidth should be expended servicing these.

In summary, enable interrupts for MII accesses and MII read errors, but not SRAM accesses.

## 6.0 Repeater MIB Support

### 6.1 REPEATER MIB SUPPORT

The following tables enumerate the groups within IEEE 802.3 u/D5, Clause 30 Repeater MIB, and the mapping to the DP83856/DP83850/DP83840 system implementation.

#### BASIC GROUP

| Repeater Managed Object Class  | Implemented In: |
|--------------------------------|-----------------|
| aRepeaterID                    | SW              |
| aRepeaterType                  | SW              |
| aRepeaterGroupCapacity         | SW              |
| aGroupMap                      | SW              |
| aRepeaterHealthState           | SW              |
| aRepeaterHealthText            | SW              |
| aRepeaterHealthData            | SW              |
| aTransmitCollisions            | DP83856         |
| acResetRepeater                | SW, DP83850     |
| acExecuteNonDisruptiveSelfTest | SW              |
| nRepeaterHealth                | SW              |
| nRepeaterReset                 | SW              |
| nGroupMapChange                | SW              |

| Group Managed Object Class | Implemented In: |
|----------------------------|-----------------|
| aGroupID                   | SW              |
| aGroupPortCapacity         | SW              |
| aPortMap                   | SW              |
| nPortMapChange             | SW              |

#### BASIC GROUP (Continued)

| Repeater Port Managed Object Class | Implemented In: |
|------------------------------------|-----------------|
| aPortID                            | SW              |
| aPortAdminState                    | SW              |
| aAutoPartitionState                | DP83850         |
| aReadableFrames                    | DP83856         |
| aReadableOctets                    | DP83856         |
| aFrameCheckSequenceErrors          | DP83856         |

| Repeater Port Managed Object Class | Implemented In: |
|------------------------------------|-----------------|
| aAlignmentErrors                   | DP83856         |
| aFramesTooLong                     | DP83856         |
| aShortEvents                       | DP83850         |
| aRunts                             | DP83856         |
| aCollisions                        | DP83850         |
| aLateEvents                        | DP83850         |
| aVeryLongEvents                    | DP83856         |
| aDataRateMismatches                | DP83856         |
| aAutoPartitions                    | DP83850         |
| alsolates                          | Plexus          |
| aSymbolErrorDuringPacket           | DP83856         |
| aLastSourceAddress                 | DP83856         |
| aPortAdminControl                  | SW              |

## 7.0 AC and DC Specifications

### 7.1 DC SPECIFICATIONS

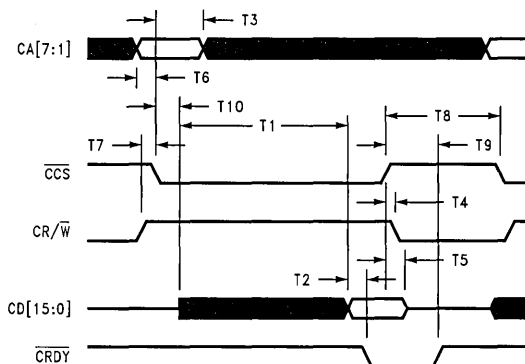
| Symbol          | Parameter                                | Conditions | Min | Max   | Units |
|-----------------|------------------------------------------|------------|-----|-------|-------|
| V <sub>OH</sub> | Minimum High Level Output Voltage        |            | 3.7 |       | V     |
| V <sub>OL</sub> | Minimum Low Level Output Voltage         |            |     | 0.4   | V     |
| V <sub>IH</sub> | Minimum High Level Input Voltage         |            | 2.0 |       | V     |
| V <sub>IL</sub> | Maximum Low Level Input Voltage          |            |     | 0.8   | V     |
| I <sub>IN</sub> | Input Current                            |            |     | ± 150 | μA    |
| I <sub>OZ</sub> | Minimum TRI-STATE Output Leakage Current |            |     | ± 160 | μA    |
| I <sub>CC</sub> | Supply Current (Calculated)              |            |     | 150   | mA    |

### 7.2 AC SPECIFICATIONS

Some timing parameters are shown more than once (both on the same timing diagram, and in different sections) for clarity.

#### 7.2.1 CPU Read Timing

| Parameter | Description                                        | Min (ns) | Max (ns) |
|-----------|----------------------------------------------------|----------|----------|
| T1        | $\overline{CSS}$ low to CPU Data valid             |          | 180      |
| T2        | CPU Data valid to $\overline{CRDY}$ low            | 10       |          |
| T3        | CPU Address hold from $\overline{CSS}$ low         | 60       |          |
| T4        | CR/ $\overline{W}$ hold from $\overline{CSS}$ high | 0        |          |
| T5        | CPU Data hold from $\overline{CSS}$ high           | 0        |          |
| T6        | CPU Address setup to $\overline{CSS}$ low          | 0        |          |
| T7        | CR/ $\overline{W}$ setup to $\overline{CSS}$ low   | 0        |          |
| T8        | $\overline{CSS}$ high between cycles               | 100      |          |
| T9        | $\overline{CSS}$ high to $\overline{CRDY}$ high    |          | 60       |
| T10       | $\overline{CSS}$ low to CPU Data driven            | 0        | 20       |

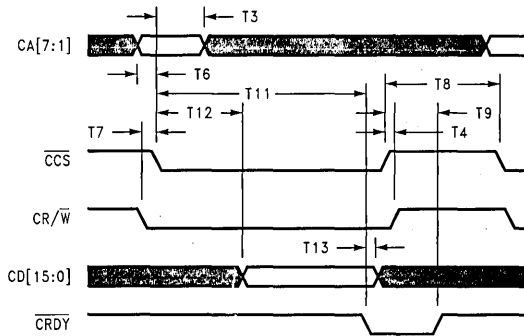


TL/F/12392-5

## 7.0 AC and DC Specifications (Continued)

### 7.2.2 CPU Write Timing

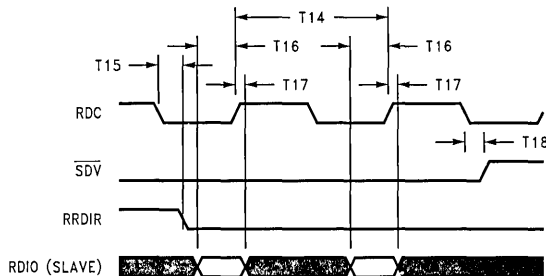
| Parameter | Description                                                      | Min (ns) | Max (ns) |
|-----------|------------------------------------------------------------------|----------|----------|
| T3        | CPU Address hold from $\overline{\text{CSS}}$ low                | 60       |          |
| T4        | CR/ $\overline{\text{W}}$ hold from $\overline{\text{CSS}}$ high | 0        |          |
| T6        | CPU Address setup to $\overline{\text{CSS}}$ low                 | 0        |          |
| T7        | CR/ $\overline{\text{W}}$ setup to $\overline{\text{CSS}}$ low   | 0        |          |
| T8        | $\overline{\text{CSS}}$ high between cycles                      | 100      |          |
| T9        | $\overline{\text{CSS}}$ high to $\overline{\text{CRDY}}$ high    |          | 60       |
| T11       | $\overline{\text{CSS}}$ low to $\overline{\text{CRDY}}$ low      |          | 180      |
| T12       | $\overline{\text{CSS}}$ low to CPU Data valid                    |          | 70       |
| T13       | CPU Data hold from $\overline{\text{CRDY}}$ low                  | 0        |          |



TL/F/12392-6

### 7.2.3 MII Slave Timing (DP83856 Receiving Data on RDIO)

| Parameter | Description                                      | Min (ns) | Typ (ns) | Max (ns) |
|-----------|--------------------------------------------------|----------|----------|----------|
| T14       | RDC pulse width                                  |          | 400      |          |
| T15       | RDC falling edge to RRDIR                        |          |          | 60       |
| T16       | RDIO setup to RDC rising edge                    | 10       |          |          |
| T17       | RDIO hold from RDC rising edge                   | 0        |          |          |
| T18       | RDC falling edge to $\overline{\text{SDV}}$ high |          |          | 60       |

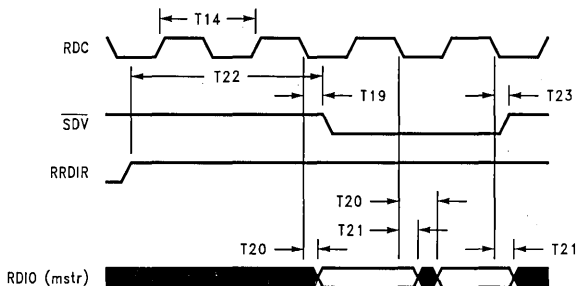


TL/F/12392-7

## 7.0 AC and DC Specifications (Continued)

### 7.2.4 MII Master Timing (DP83856 Sending Data on RDIO)

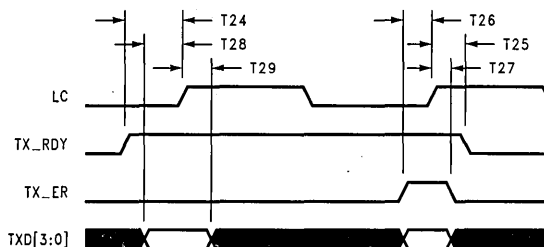
| Parameter | Description                                        | Min (ns) | Typ (ns) | Max (ns) |
|-----------|----------------------------------------------------|----------|----------|----------|
| T14       | RDC pulse width                                    |          | 400      |          |
| T19       | RDC falling edge to $\overline{SDV}$ falling edge  |          |          | 60       |
| T20       | RDC falling edge to RDIO valid                     |          |          | 60       |
| T21       | RDC falling edge to RDIO invalid                   | 0        |          |          |
| T22       | RRDIR rising edge to $\overline{SDV}$ falling edge | 600      |          |          |
| T23       | RDC falling edge to $\overline{SDV}$ rising edge   |          |          | 60       |



TL/F/12392-8

### 7.2.5 TX Bus Timing

| Parameter | Description                       | Min (ns) | Typ (ns) | Max (ns) |
|-----------|-----------------------------------|----------|----------|----------|
| T24       | TX_RDY setup to LC rising edge    | 9        |          |          |
| T25       | TX_RDY hold from LC rising edge   | 3        |          |          |
| T26       | TX_ER setup to LC rising edge     | 5        |          |          |
| T27       | TX_ER hold from LC rising edge    | 2        |          |          |
| T28       | TXD[3:0] setup to LC rising edge  | 6        |          |          |
| T29       | TXD[3:0] hold from LC rising edge | 2        |          |          |

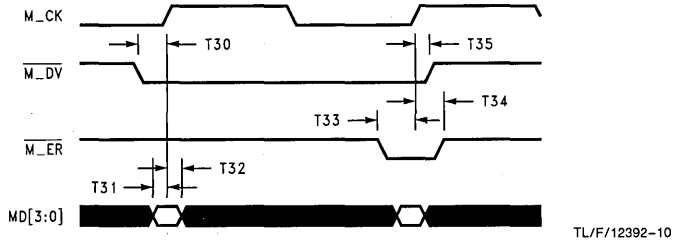


TL/F/12392-9

## 7.0 AC and DC Specifications (Continued)

### 7.2.6 Management Bus Timing

| Parameter | Description                                   | Min (ns) | Typ (ns) | Max (ns) |
|-----------|-----------------------------------------------|----------|----------|----------|
| T30       | $\overline{M\_DV}$ setup to M_CK rising edge  | 3        |          |          |
| T31       | MD[3:0] setup to M_CK rising edge             | 1        |          |          |
| T32       | MD[3:0] hold from M_CK rising edge            | 0        |          |          |
| T33       | $\overline{M\_ER}$ setup to M_CK rising edge  | 3        |          |          |
| T34       | $\overline{M\_ER}$ hold from M_CK rising edge | 1        |          |          |
| T35       | $\overline{M\_DV}$ hold from M_CK rising edge | 1        |          |          |



### 7.2.7 SRAM Read Timing

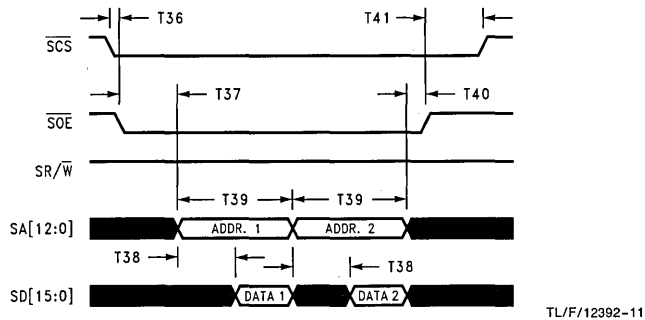
| Parameter    | Description                                       | Min (ns) | Typ (ns) | Max (ns) |
|--------------|---------------------------------------------------|----------|----------|----------|
| T36          | $\overline{SCS}$ low to $\overline{SOE}$ low      | 0        |          |          |
| T37 (Note 1) | $\overline{SOE}$ low to SA[12:0] valid            |          | 20       | 30       |
| T38 (Note 2) | SA[12:0] valid to SD[15:0] valid (SRAM $t_{su}$ ) |          |          | 25       |
| T39 (Note 3) | SA[12:0] width                                    | 50       |          |          |
| T40 (Note 4) | SA[12:0] invalid to $\overline{SOE}$ high         | 0        |          |          |
| T41          | $\overline{SOE}$ high to $\overline{SCS}$ high    | 35       |          |          |

**Note 1:** All SRAM read cycles are Address controlled.

**Note 2:** SRAM must have a read access time of 20 ns or faster.

**Note 3:** The DP83856 latches data prior to changing the SA[12:0] value.

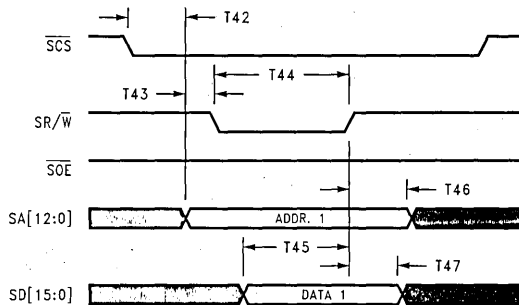
**Note 4:** The DP83856 latches data prior to terminating SOE.



## 7.0 AC and DC Specifications (Continued)

### 7.2.8 SRAM Write Timing

| Parameter | Description                                 | Min (ns) | Typ (ns) | Max (ns) |
|-----------|---------------------------------------------|----------|----------|----------|
| T42       | $\overline{SCS}$ low to SA[12:0] valid      | 30       |          |          |
| T43       | SA[12:0] valid to SR/ $\overline{W}$ low    | 10       |          |          |
| T44       | SR/ $\overline{W}$ width                    | 35       |          |          |
| T45       | SD[15:0] valid to SR/ $\overline{W}$ high   | 25       |          |          |
| T46       | SR/ $\overline{W}$ high to SA[12:0] invalid | 15       |          |          |
| T47       | SR/ $\overline{W}$ high to SD[15:0] invalid | 15       |          |          |

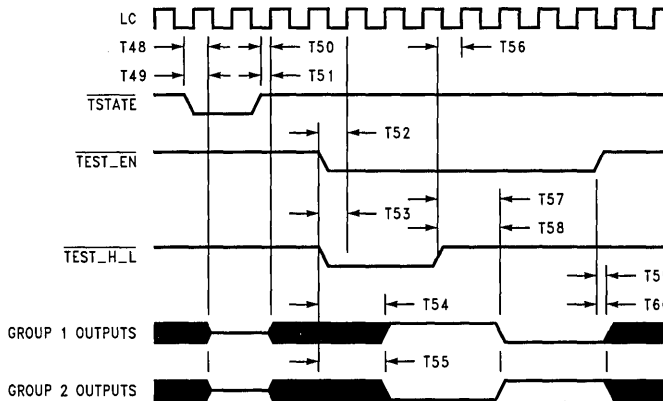


TL/F/12392-12

## 7.0 AC and DC Specifications (Continued)

### 7.2.9 Test Mode Timing

| Parameter | Description                                                                                   | Min (ns) | Typ (ns) | Max (ns)                     |
|-----------|-----------------------------------------------------------------------------------------------|----------|----------|------------------------------|
| T48       | $\overline{\text{TSTATE}}$ low to Group 1 Outputs Hi-Z                                        |          |          | 25                           |
| T49       | $\overline{\text{TSTATE}}$ low to Group 2 Outputs Hi-Z                                        |          |          | 25                           |
| T50       | $\overline{\text{TSTATE}}$ high to Group 1 Outputs driven                                     |          |          | 25                           |
| T51       | $\overline{\text{TSTATE}}$ high to Group 2 Outputs driven                                     |          |          | 25                           |
| T52       | $\overline{\text{TEST\_EN}}$ low setup to LC rising edge                                      | 20       |          |                              |
| T53       | $\overline{\text{TEST\_H\_L}}$ low setup to LC rising edge                                    | 20       |          |                              |
| T54       | $\overline{\text{TEST\_EN}}$ , $\overline{\text{TEST\_H\_L}}$ low to Group 1 Outputs high     |          |          | $2 * \text{LC} + 0\text{ns}$ |
| T55       | $\overline{\text{TEST\_EN}}$ , $\overline{\text{TEST\_H\_L}}$ low to Group 2 Outputs low      |          |          | $2 * \text{LC} + 0\text{ns}$ |
| T56       | $\overline{\text{TEST\_H\_L}}$ high setup to LC rising edge                                   | 20       |          |                              |
| T57       | $\overline{\text{TEST\_EN}}$ low, $\overline{\text{TEST\_H\_L}}$ high to Group 1 Outputs low  |          |          | $2 * \text{LC} + 0\text{ns}$ |
| T58       | $\overline{\text{TEST\_EN}}$ low, $\overline{\text{TEST\_H\_L}}$ high to Group 2 Outputs high |          |          | $2 * \text{LC} + 0\text{ns}$ |
| T59       | $\overline{\text{TEST\_EN}}$ high to Group 1 Outputs undefined                                |          |          | $2 * \text{LC} + 0\text{ns}$ |
| T60       | $\overline{\text{TEST\_EN}}$ high to Group 2 Outputs undefined                                |          |          | $2 * \text{LC} + 0\text{ns}$ |



TL/F/12392-13

Group 1 output pin numbers are:

21, 23, 27, 31, 35, 37, 39, 43, 61, 64, 68, 70, 72, 74, 78, 80, 82, 87, 89, 90, 92, 96, 98, 100, 104, 108, 113, 115.

Group 2 output pin numbers are:

20, 22, 24, 30, 34, 36, 38, 42, 44, 65, 69, 71, 73, 75, 79, 81, 83, 88, 91, 93, 97, 99, 103, 105, 114, 116.



# An Introduction to Auto-Negotiation

National Semiconductor  
Application Note 986  
Bill Bunch



AN-986

## Contents

- 1.0 INTRODUCTION
- 2.0 WHAT IS AUTO-NEGOTIATION?
- 3.0 BENEFITS OF AUTO-NEGOTIATION
- 4.0 ARCHITECTURE
- 5.0 EXPANDABILITY
- 6.0 CONCLUSION
- 7.0 REFERENCES
- 8.0 DEFINITIONS

## 1.0 Introduction

NWay Auto-Negotiation is a technology which was introduced by National Semiconductor to the IEEE 802.3u 100BASE-T working group in the Spring of 1994 as a result of the need for a mechanism to accommodate multi-speed network devices. National's NWay technology was chosen as the basis for this mechanism due to its simplicity, low cost, flexibility, interoperability with the installed base, and adaptability to future technologies. Currently, the Auto-Negotiation mechanism is defined in Clause 28 of the D5 draft of the ANSI/IEEE Std 802.3 MAC Parameters, Physical Layer, Medium Attachment Units and Repeater for 100 Mb/s Operation. This draft has been approved by the IEEE 802.3 Working Group. Refer to section 8.0 for definitions used in this document.

## 2.0 What is Auto-Negotiation?

Auto-Negotiation is a mechanism that takes control of the cable when a connection is established to a network device. Auto-Negotiation detects the various modes that exist in the device on the other end of the wire, the Link Partner, and

advertises its own abilities to automatically configure the highest performance mode of interoperation. As a standard technology, this allows simple, automatic connection of devices that support a variety of modes from a variety of manufacturers.

Auto-Negotiation acts like a rotary switch that automatically switches to the correct technology, such as 10BASE-T, 100BASE-TX, 100BASE-T4, or a corresponding Full Duplex mode. Once the highest performance common mode is determined, Auto-Negotiation passes control of the cable to the appropriate technology and becomes transparent until the connection is broken.

Auto-Negotiation leverages the proven link function of 10BASE-T to provide robust operation over Category 3, 4, or 5 Unshielded Twisted Pair (UTP.)

### 2.1 BASIC OPERATION

There are two basic cases that Auto-Negotiation accounts for as shown in *Figure 1*:

1. Auto-Negotiation exists at both ends of a twisted-pair link. (Node A to Hub)
2. Auto-Negotiation exists at only one end of a twisted-pair link. (Node B to Hub)

Auto-Negotiation is most useful if it exists at both ends of the link since both ends speak the same "language" at start up. This allows a rich set of information to be transferred.

The key to Auto-Negotiation's interoperation with installed, legacy LANs is the Parallel Detection function. The Parallel Detection function accounts for the case where only one end of a twisted-pair link has Auto-Negotiation. For example, consider an installed 10BASE-T node connected to a hub that supports 10BASE-T, 100BASE-TX, and Auto-Negotiation (see *Figure 1*). In this case, the hub recognizes the unique signals that the 10BASE-T only device produces and switches to 10BASE-T operation.

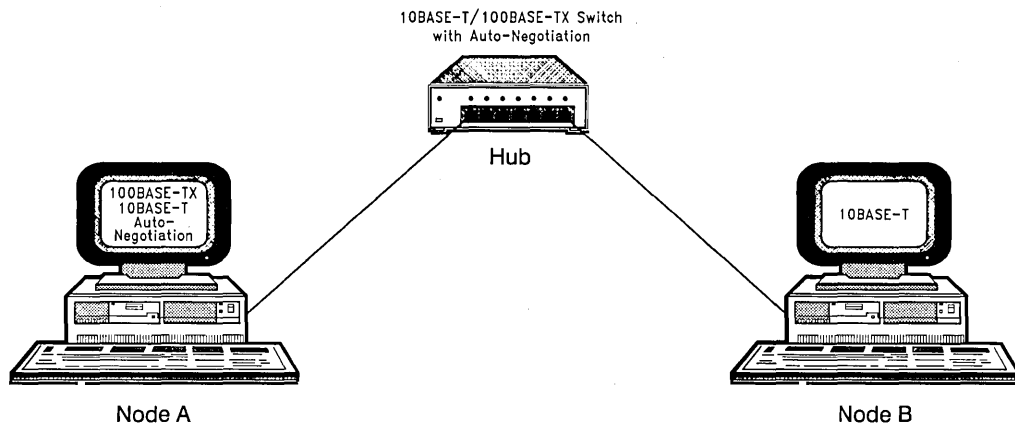


FIGURE 1. Connection Example

TL/F/12389-1

## 2.2 OPTIONAL OPERATION

In addition to the basic connection mechanism, Auto-Negotiation also provides the following optional additional features:

### 2.2.1 Management Interface

The serial management interface of the Media Independent Interface (MII) register set provides a mechanism for additional control of Auto-Negotiation. It also provides a means to gather network status information.

### 2.2.2 Next Page Function

After exchanging the Base Page, which contains the information to make a connection automatically, if both ends of the link indicate support for the Next Page function, additional data may be exchanged. This allows extensions to the standard and proprietary extensions to exist without affecting interoperability.

### 2.2.3 Remote Fault Indication

The basic transport mechanism for simple fault information is built into Auto-Negotiation, but the detection and advertisement of any particular fault is not required. Remote Fault Indication allows a device that is able to detect faults (e.g., wrong cable type, wiring fault, etc.) to advertise the presence of the fault to the Link Partner.

The Remote Fault Indication may be used in conjunction with the Next Page function to transfer more information about the type of fault that occurred.

## 3.0 Benefits of Auto-Negotiation

### 3.1 AUTOMATIC CONNECTION

The primary benefit of Auto-Negotiation is the automatic connection of the highest performance technology available without any intervention from a user, manager, or management software.

### 3.2 BACKWARDS COMPATIBILITY

If Auto-Negotiation exists at only one end of a twisted-pair link, it determines that the Link Partner does not support the Auto-Negotiation mechanism. Instead of exchanging configuration information, it examines the signal it is receiving. If Auto-Negotiation discovers that the signal matches a technology that the device supports, it will automatically connect that technology. This function, known as Parallel Detection, gives Auto-Negotiation the ability to be compatible with any device that does not support Auto-Negotiation, yet supports: 10BASE-T, 100BASE-TX, or 100BASE-T4. Connection to any technology via Parallel Detection other than those listed above is not supported by Auto-Negotiation.

### 3.3 NETWORK PROTECTION

In the event that no common technology exists, Auto-Negotiation will not make a connection. This ensures preservation of network integrity and minimization of network down time.

In particular, Hubs are a primary beneficiary of this feature. For example, if a user connects a 100BASE-T4 device into a 10BASE-T/100BASE-TX switch, the result could be catastrophic for all the users connected through that switch. However, if the hub has Auto-Negotiation, it would refuse the connection and allow the rest of the network to proceed as usual. In fact, with Auto-Negotiation in the hub, the network users are protected from any connection that the hub cannot recognize or accept.

## 3.4 TECHNOLOGY EXTENSIONS

If Auto-Negotiation exists on both ends of a twisted-pair link, then both ends advertise their abilities to the other. Auto-Negotiation incorporates a robust handshake that ensures data integrity. The devices compare their abilities and connect at the highest performance common technology shared.

Auto-Negotiation has been defined for flexibility. Standard technologies can use the basic Auto-Negotiation logic with their own definitions for the information to be exchanged (see section 5.0 for details). Currently, IEEE 802.3 and 802.9 Working Groups each have their own, independent codes which allows the technologies to define which abilities can be advertised; In total, 32 of these codes can exist. IEEE 802.3 currently supports: 10BASE-T, 10BASE-T Full Duplex, 100BASE-TX, 100BASE-TX Full Duplex, and 100BASE-T4. Even within the IEEE 802.3 code space there is room for future technologies or enhancements.

### 3.5 UPGRADE PATH

New nodes on the market will have 100Mb/s functionality as well as the traditional 10BASE-T. This means that there will be some latent performance available as these new nodes are added to an old 10BASE-T network. When the performance issue becomes critical, the latent ability can be tapped into by upgrading the hub. Auto-Negotiation enables the upgrade to occur without reconfiguring each node and/or each port on the new hub.

### 3.6 MANAGEMENT INTERFACE

While no management intervention is required for automatic connection, a management interface has been provided to give optional control and status of Auto-Negotiation. The management interface provides the following capabilities:

1. Determine why a connection was refused
2. Determine which abilities exist on the network
3. Change connection speed
4. Retrieve fault status
5. Exchange arbitrary configuration information with a Link Partner (in conjunction with the Next Page function)

These capabilities are useful in a managed-hub application since they give the manager remote access to all the above information and control. These functions are useful for node solutions with Auto-Negotiation as well. However, in the case of a node, the information is only available to the user of that node and not to the network at large. This information would be useful in installation and diagnostic software to help guide the user in resolving any difficulties.

### 3.7 Proprietary Extension

Auto-Negotiation has the option to send additional pieces of information after the "base" negotiation that determines the network connection before enabling the data service. This is known as the Next Page function. Among other things, it can be used to send information that corresponds to an Organizationally Unique Identifier so that extra features could be implemented on a proprietary basis, yet not conflict with standard operation. Both ends of a twisted-pair link must have Auto-Negotiation with support for the Next Page function in order to take advantage of this feature.

Specific remote fault type information transfer can also be supported using this flexible mechanism.

## 4.0 Architecture

To support the many different technologies that are on the market today or will be available in the future, Auto-Negotiation has been architected in a way that provides extensibility and flexibility.

Basically, an Auto-Negotiation device advertises its abilities and detects the abilities of the remote device that it is connected to, known as the Link Partner. Once Auto-Negotiation has received the Link Partner's abilities in a robust manner and it receives acknowledgment that its abilities have also been received by the Link Partner, Auto-Negotiation compares the two sets of abilities and decides which technology to connect. This decision is based upon a pre-agreed priority of technologies. Auto-Negotiation attaches the highest performance common technology to the medium and becomes transparent until the link goes down or is reset.

### 4.1 ABILITY TRANSPORT MECHANISM

The basic mechanism that Auto-Negotiation uses to advertise a device's abilities is a series of link pulses which encode a 16 bit word, known as a Fast Link Pulse (FLP) Burst. An FLP Burst is composed of 17 to 33 link pulses which are identical to the link pulses used in 10BASE-T to determine whether a link has a valid connection (sometimes referred to as Normal Link Pulses or NLPs). FLP Bursts occur at the same interval as NLPs,  $16 \pm 8$  ms. An FLP Burst has a nominal duration of 2 ms. *Figure 2* shows the nominal timing of FLP Bursts.

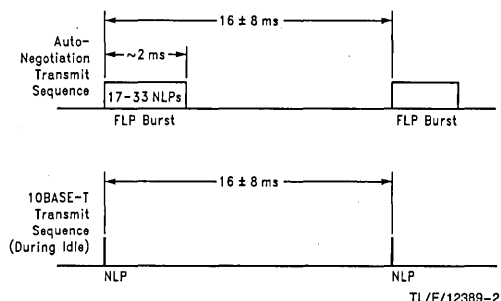
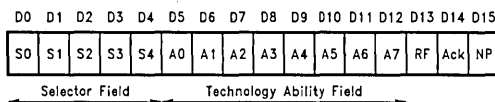


FIGURE 2. FLP Burst Timing

An FLP Burst interleaves clock pulses with data pulses to encode a 16 bit word. The absence of a pulse within a time window following a clock pulse encodes a logic zero and a pulse within the time window following a clock pulse encodes a logic one.

### 4.2 DATA ENCODING

The key to Auto-Negotiation's flexibility and expandability is the encoding of the 16 bit word. The 16 bit word is referred to as the Link Code Word (LCW). The LCW is encoded as shown in *Figure 3*.



TL/F/12389-3

FIGURE 3. Base Link Code Word Encoding

The Selector Field, S[4:0], allows 32 different definitions of the Technology Ability Field to coexist. The intention is to allow standard technologies to leverage the basic Auto-Negotiation mechanism. Currently, S[4:0] = <00001> is assigned to IEEE 802.3 and S[4:0] = <00010> is assigned to IEEE 802.9. Two more codes are reserved for expansion of Auto-Negotiation. The remaining codes are reserved to be assigned to standard technologies that wish to leverage this mechanism, yet fall outside the scope of the currently defined Selector Field values.

The Technology Ability Field, A[7:0], is defined relative to the Selector Field value of the Link Code Word. For IEEE 802.3 there are bits defined to advertise:

- 1 - 100BASE-TX Full Duplex
- 2 - 100BASE-T4
- 3 - 100BASE-TX
- 4 - 10BASE-T Full Duplex
- 5 - 10BASE-T

The above list also defines the priority hierarchy for resolving multiple common abilities. That is, if both devices support both 10BASE-T and 100BASE-TX, Auto-Negotiation at both ends will connect 100BASE-TX instead of 10BASE-T.

Priority resolution works such that when the 3 remaining bits in the Technology Ability Field are eventually defined, the new technology can be inserted anywhere in the list without disturbing the existing hierarchy. This means that the 3 reserved bits can be assigned without causing interoperability problems with any Auto-Negotiation device produced before these bits were defined.

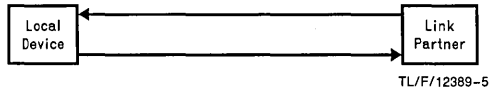
The Remote Fault bit, RF, allows transmission of simple fault information to the Link Partner.

The Acknowledge bit, Ack, is used by the synchronization mechanism to ensure robust data transfer.

The Next Page bit, NP, advertises to the Link Partner whether the Next Page function is supported. The Next Page function is used to send additional information beyond the basic configuration information. Both ends must have this ability in order to exchange this type of information.

### 4.3 AUTO-NEGOTIATION SYNCHRONIZATION

Auto-Negotiation must ensure that the Link Partner receives the Link Code Word correctly and that the Link Partner's Link Code Word is received correctly in order to make a connection decision. Auto-Negotiation uses the Arbitration function to accomplish this. *Figure 4* illustrates the following example.



TL/F/12389-5

- ① Transmit LCW[LD], Ack = 0
- ② Receives same LCW[LP] 3 times (ignore Ack). Transmits LCW[LD], Ack = 1. (Local Device has received LCW[LD] OK)
- ③ Receives same LCW[LP] 3 times w/ Ack = 1. (Link Partner has received LCW OK)
- ④ Transmit LCW[LD], Ack = 1 6-8 more times.
- ⑤ Stop transmitting LCW[LD]. Configure highest performance common technology.

Note: Process is symmetric (i.e. both the Local Device and the Link Partner follow the same procedure.)

**FIGURE 4. Synchronization Example**

The Local Device begins by transmitting its Link Code Word, (LCW[LD]), with the Ack bit not set. Once 3 consecutive, matching Link Code Words are received from the Link Partner (LCW[LP]), (ignoring Ack), the Local Device sets the Ack bit in the transmitted Link Code Word to indicate that it has received the Link Partner's Link Code Word correctly.

The Local Device continues transmitting its Link Code Word. Upon receiving 3 consecutive, matching Link Code Words from the Link Partner with the Ack bit set, the Local Device knows that the Link Partner has also received the Link Code Word correctly. The Local Device transmits the Link Code Word with the Ack bit set 6-8 additional times to ensure that a complete handshake has taken place.

Now, both the Local Device and the Link Partner have exchanged their base Link Code Words. Each device compares their abilities and the highest performance common technology as determined by priority resolution is connected to the medium.

#### 4.4 PARALLEL DETECTION

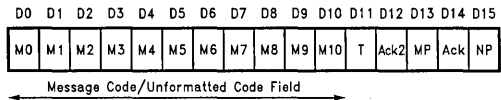
To account for technologies that existed prior to Auto-Negotiation, Auto-Negotiation passes the signals present on the receiver to the 100BASE-TX and 100BASE-T4 Link Monitor functions. If Auto-Negotiation determines that exactly one Link Monitor function indicates that the link is good, then it can connect that technology to the media. Note, however, that this function is only implemented for 10BASE-T, 100BASE-TX, and 100BASE-T4. Future multi-mode devices will use Auto-Negotiation as the basis of automatic mode switching.

Auto-Negotiation incorporates a modified 10BASE-T Link Integrity Test function in order to interoperate properly with installed 10BASE-T devices. The modifications ensure that Auto-Negotiation can control the function such that 10BASE-T devices are always correctly detected.

#### 4.5 NEXT PAGE FUNCTION

If the Next Page bit is set in both the outgoing and incoming Link Code Words, then both the Local Device and the Link Partner are able to support the Next Page function and will participate in Next Page exchange. Once the first Link Code Word has been exchanged, both sides have the information required to configure the highest common technology. However, if Next Page exchange occurs then Auto-Negotiation does not configure the highest common technology until Next Page exchange has completed.

Next Page exchange works in the same way that the "base" Link Code Words were exchanged. The main difference is the encoding of the exchanged Link Code Words which is shown in *Figure 5*.



TL/F/12389-4

**FIGURE 5. Next Page Link Code Word Encoding**

The Next Page bit, NP, indicates that an additional Next Page will be exchanged.

The Acknowledge, Ack bit works the same as for the base Link Code Word exchange.

Message Page, MP, indicates whether the Message Code Field, M[10:0], will be interpreted as a Message Code or an Unformatted Code. Message Codes are pre-defined messages in the IEEE 802.3 standard, Clause 28. Unformatted Codes are arbitrary pieces of data. Following a base Link Code Word exchange with the IEEE 802.3 Selector Field value, Unformatted Codes follow Message Codes with information required by the Message Code.

There are two different ways of interpreting a received Next Page. If the Message Page bit is set, then the Message Code Field, M[10:0], is a binary code that corresponds to a pre-defined message in the IEEE 802.3 standard, Clause 28. There are 2048 possible message codes. Of these, 8 codes are defined (all other codes are undefined at present): 2 codes are reserved for Auto-Negotiation expansion and the remaining 6 codes are defined as follows:

Null Message: Code exchanged if there is no further information to be transmitted while the Link Partner is still transmitting information.

One Unformatted Page containing a Technology Ability Field follows: Provides extension of the base Link Code Word.

Two Unformatted Pages containing Technology Ability Field information follows: Provides extension of the base Link Code Word.

One Unformatted Page with a binary encoded Remote Fault follows: Unformatted Page contains Remote Fault type; Remote Fault Test, Link Loss, Jabber, or Parallel Detection Fault

OUI Tagged Message: Organizationally Unique Identifier followed by one Unformatted Page (defined by the transmitting organization).

PHY ID Tagged Message: PHY ID followed by one Unformatted Page (defined by the transmitting organization).

The Acknowledge 2 bit, Ack2, is set by the receiving device to indicate that the device supports the function indicated by the message.

The Toggle bit, T, is set by the Arbitration function within Auto-Negotiation to ensure proper synchronization with the Link Partner during Next Page exchange.

#### 4.6 REMOTE FAULT SENSING

##### 4.6.1 Simple Remote Fault Transport Mechanism

A basic remote fault status transport mechanism is built into the Auto-Negotiation function (i.e., mandatory). However, the ability to sense and categorize fault types is not required. To transfer simple remote fault status, a device which has detected a remote fault will set the Remote Fault bit in the Auto-Negotiation Advertisement Register (ANAR), and renegotiate. This will advertise to the Link Partner that a remote fault has been detected. If negotiation subsequently completes, the Remote Fault bit in the ANAR will be reset to clear the fault condition.

Upon detection of the Remote Fault bit in the Auto-Negotiation Link Partner Advertisement Register (ANLPAR), the device will set the Remote Fault bit in the MII status register.

**Note:** All registers are defined as part of the MII register set.

#### 4.6.2 Simple Remote Fault Sensing

Devices may implement any remote fault detection mechanism desired and use this transport mechanism to inform the Link Partner of a fault. The meaning of the fault to the receiver is limited, however. Reception of remote fault status only informs a device that something is wrong with the link rather than specifying the type of fault that has occurred.

As an example, a device could detect a fault as follows. If a device is attempting to Auto-Negotiate yet "never" receives a valid set of signals that will allow it to connect, management software could detect this as being caused by a fault in making a connection. The device could then set the Remote Fault bit in the ANAR and renegotiate. The scenario described above could be caused by: (see *Figure 6*)

1. The Local Device has a fault in the wiring of the receive pair.

The Link Partner would have received the remote fault information and set the status bit informing management that a fault has occurred.

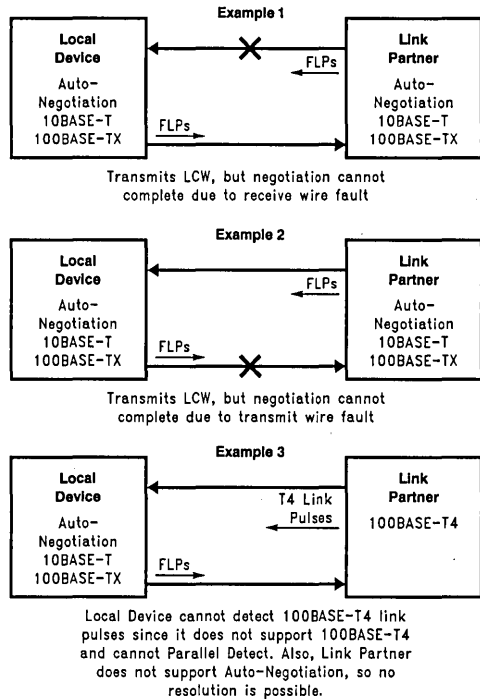
2. The Local Device has a fault in the wiring of the transmit pair.

The Link Partner could never receive the remote fault information. If the Link Partner also supported this type of remote fault sensing, then the situation would be equivalent to example 1, where the Local Device would inform management of the fault status.

In this case, the Local Device will detect that the Link Partner is Auto-Negotiation able and set its outgoing Ack bit. The Local Device will "never" receive Ack set from the Link Partner. Since the Local Device's management agent knows that both devices are Auto-Negotiating, but cannot complete since there is no acknowledgment from the Link Partner, there must be something wrong with the transmission path.

3. The Link Partner is not transmitting FLP Bursts and instead transmits signals of a technology that the Local Device cannot support.

Since the Link Partner does not support Auto-Negotiation, the remote fault information is not received by the Link Partner. Note that no connection should be allowed since there are no common technologies between the devices. The Local Device will continue to send link pulses indefinitely. Software may determine that a fault continues to persist and notify any local management agent.



TL/F/12389-6

**FIGURE 6. Remote Fault Scenarios**

#### 4.6.3 Specific Remote Faults via Next Page

It is possible for Auto-Negotiation to complete, even though some type of remote fault is present that can be detected. For example, a device may be jabbering, the wire may not support the 100Mb/s technology, or there is excessive noise present.

While this type of fault could be transferred using the simple Remote Fault transport mechanism, it may be beneficial to inform the Link Partner which type of fault is being experienced. This can be accomplished if both ends of the link participate in a Next Page exchange to transfer the fault type information. The wire connection must be such that an Auto-Negotiation page exchange can complete.

## 5.0 Expandability

Auto-Negotiation has been architected to provide extensive code space that will allow the basic mechanism to be leveraged and remain interoperable regardless of the nature of new technologies.

### 5.1 AUTO-NEGOTIATION ON ALTERNATE MEDIA

Auto-Negotiation is easily adaptable to virtually any technology that uses twisted pair wiring. While not standardized, the same mechanism could be used over media types other than twisted pair by replacing the encoding method with one that is compatible with the given media. For example, since link pulses do not directly translate onto fiber, an alternate coding scheme could be defined to replace the link pulses. The algorithm and Link Code Word encodings would all remain the same.

### 5.2 NEXT PAGE EXTENSION

The Next Page function is architected to provide virtually unlimited code space. The Message Code space has 2040 codes that may be defined. Implementations need only consider what is an acceptable time to make a connection.

#### 5.2.1 Technology Ability Field Extension

Within a given Selector Field, the base page has enough space for 8 different technologies (assuming they are to be advertised independently). If all of the base page bits are defined, the Next Page function can be used to extend this to support additional technologies. Thus far, codes have been reserved to support up to 16 additional bits dedicated to providing technology information.

#### 5.2.2 Proprietary Extension

The Next Page function also provides the flexibility for manufacturers to define any additional information that may be used to provide control and/or status to a management agent.

### 5.3 NETWORK TYPE EXTENSION

Through the Selector Field code space, 30 fundamentally different network types can be accommodated by the Auto-Negotiation function. Currently, IEEE 802.3, CSMA/CD LANs, and IEEE 802.9 Integrated Services LAN have adopted Auto-Negotiation and reside in this code space. Token Ring, Wireless, and others could conceivably leverage all or part of Auto-Negotiation to provide a greater level of interoperability.

## 6.0 Conclusion

Auto-Negotiation is a standard, simple, low cost, flexible mechanism for providing connection interoperability between IEEE 802.3 LANs. Auto-Negotiation forms the basis for a highest common performance link configuration mechanism. In addition, Auto-Negotiation provides management control and is a valuable network status tool. Auto-Negotiation's simplicity facilitates implementing cost effective multi-function nodes and/or hubs. Auto-Negotiation's flexible architecture ensures that future technology interoperability needs can be met.

National Semiconductor provided its NWay technology and expertise to create Clause 28 of ANSI/IEEE Std 802.3u Draft D5 which embodies the Auto-Negotiation Function. This draft specifically supports configuring the highest performance common mode between 10BASE-T and 100BASE-T devices, enabling multi-vendor, standard interoperability a reality for IEEE 802.3 compatible LANs.

## 7.0 References

International Standard ISO/IEC 8802-3: 1992, 3rd. ed., ANSI/IEEE Std 802.3  
 IEEE Std 802.3u/D5-1995 (Draft supplement to ISO/IEC 8802-3:1993 ANSI/IEEE Std 802.3-1993 ed.)

## 8.0 Definitions

**Attachment Unit Interface (AUI).** In 10BASE-T, the interface between the MAU and the DTE within a data station.

**ability.** A mode which a device can advertise using Auto-Negotiation.

**advertised ability.** An operational mode that is advertised using Auto-Negotiation.

**Auto-Negotiation.** The function which allows two devices at either end of a link segment to negotiate common data service functions.

**Base Link Code Word.** The first 16-bit message exchanged during Auto-Negotiation.

**Base Page.** See Base Link Code Word.

**Data Terminal Equipment (DTE).** Any source or destination of data connected to the LAN.

**Fast Link Pulse (FLP) Burst.** A group of no more than 33 and not less than 17 10BASE-T compatible link integrity test pulses. Each FLP Burst encodes 16 bits of data using an alternating clock and data pulse sequence.

**Full Duplex.** A type of networking which supports simultaneous reception and transmission.

**jabber.** A condition wherein a station transmits for a period of time longer than permissible, usually due to a fault condition.

**link.** The transmission path between any two interfaces of generic cabling.

**Link Code Word.** The 16 bits of data encoded into a Fast Link Pulse Burst.

**Link Partner.** The device at the opposite end of a link segment from the local device. The Link Partner device may be either a DTE or repeater.

**link pulse.** Communication mechanism used in 10BASE-T and 100BASE-T networks to indicate link status and (in Auto-Negotiation equipped devices) to communicate information about abilities and negotiate communication methods. 10BASE-T uses Normal Link Pulses (NLPs), which indicate link status only. 10BASE-T and 100BASE-T devices equipped with Auto-Negotiation exchange information using a Fast Link Pulse mechanism which is compatible with 10BASE-T.

**link segment.** The point-to-point full duplex medium connection between two and only two Medium Dependent Interfaces (MDIs).

**local ability.** See ability. Relative to the Local Device.

**Local Device.** The local station which may attempt to Auto-Negotiate with a Link Partner. The Local Device may be either a DTE or repeater.

**Medium Attachment Unit (MAU).** A device containing an AUI, PMA, and MDI, used to connect a repeater or DTE to a transmission medium.

**Medium Dependent Interface.** The mechanical and electrical interface between the transmission medium and the MAU (10BASE-T) or PHY (100BASE-T).

**Media Independent Interface (MII).** A signal interface which maps to MAC service definitions.

**Message Code.** The pre-defined 11-bit code contained in an Auto-Negotiation Message Page.

**Message Page.** An Auto-Negotiation Next Page encoding which contains a pre-defined 11-bit message code.

**Next Page function.** The algorithm which governs Next Page communication.

**Next Page bit.** A bit in the Auto-Negotiation Base Link Code Word that indicates there are additional Link Code Words with Next Pages to be exchanged.

**NLP Receive Link Integrity Test function.** Auto-Negotiation link integrity test function which allows backward compatibility with the 10BASE-T Link Integrity Test function (See *Figure 14-6* in IEEE 802.3).

**NLP sequence.** A Normal Link Pulse sequence, as defined in IEEE 802.3 section 14.2.1.1.

**Normal Link Pulse (NLP).** An out-of-band communications mechanism used in 10BASE-T to indicate link status.

**Physical Layer Device (PHY).** The portion of the physical layer between the MDI and MII.

**Physical Medium Attachment (PMA) sublayer.** The portion of the physical layer that contains the functions for transmission, collision detection, reception, and (in the case of 100BASE-T4) clock recovery and skew alignment.

**page.** In Auto-Negotiation, the encoding for a Link Code Word. Auto-Negotiation can support multiple Link Code Word encodings. The base page has a constant encoding as defined in IEEE 802.3u D4, section 28.2.1.2. Additional pages may have a pre-defined encoding (see Message Page) or may be custom encoded (see Unformatted Page.)

**parallel detection.** In Auto-Negotiation, the ability to detect 100BASE-TX and 100BASE-T4 technology specific link signalling while also detecting the NLP sequence of a 10BASE-T device.

**Priority Resolution function.** The mechanism used by Auto-Negotiation to select the network connection type where more than one common network ability exists (100BASE-TX, 100BASE-T4, 10BASE-T, etc). The priority resolution table defines the relative hierarchy of connection types from the highest performance to the lowest performance.

**remote fault.** The generic ability of a Link Partner to signal its status even in the event that it may not have an operational link.

**renegotiation.** Re-start of the Auto-Negotiation function caused by a management or user interaction.

**segment.** The medium connection, including connectors, between MDIs in a CSMA/CD LAN.

**Selector Field.** A 5 bit field in the base Link Code Word encoding that is used to encode up to 32 types of messages which define basic abilities.

**Technology Ability Field.** An 8 bit field in the Auto-Negotiation base Link Code Word encoding that is used to indicate the abilities of a Local Device, such as support for 10BASE-T, 100BASE-TX, 100BASE-T4, as well as Full Duplex capabilities.

**Unformatted Page.** A Next Page encoding which contains an unformatted 11-bit message field. Use of this field is defined through Message Codes and information contained in the Unformatted Page message field.

# 100BASE-TX Unmanaged Repeater Design Recommendations

National Semiconductor  
Application Note 1010  
Todd Vafiades



## 1.0 INTRODUCTION

This application note provides the information necessary to design an unmanaged 12-port 100BASE-TX repeater based on National Semiconductor's DP83850, DP83840, and DP83223 integrated circuits. The DP83850 is a full featured Repeater Interface Controller (RIC™) capable of supporting up to 12 100BASE-X ports. The DP83840 PHY device with the DP83223 twisted pair transceiver combine to provide the 100BASE-TX compliant Physical Layer and Physical Medium Dependent sublayer.

A design based on these three devices allows for a simple, low cost 12-port 100 Mb/s repeater solution.

While considerations such as Auto-Negotiation and 10/100 Mb/s operation are noted herein, detailed emphasis is placed on the fundamental design requirements, from the MDI (Medium Dependent Interface) to the MII (Medium

Independent Interface), for an unmanaged 100BASE-TX repeater. System design aspects such as interconnection, clock distribution and physical layout are provided.

It is recommended that this application note be reviewed in conjunction with the latest version datasheets for the DP83850, DP83840, and DP83223 devices.

## 2.0 OVERVIEW

The block diagram in *Figure 1* illustrates the interconnection and layout for a 12-port 100BASE-TX repeater. Although the basic functionality of a 100BASE-TX repeater is similar to that of a 10BASE-T repeater, there are some important differences. These differences include data rate, signal encoding/decoding, and link integrity verification.

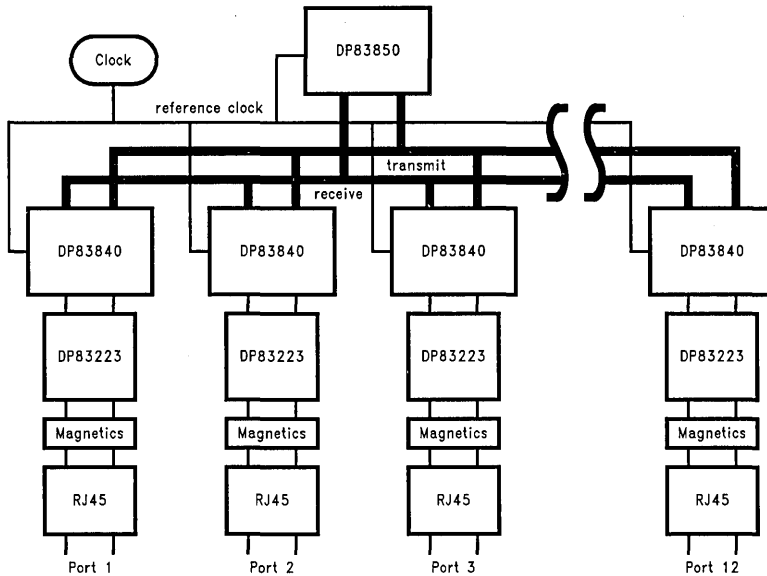


FIGURE 1. 100BASE-TX Repeater

TL/F/12505-1



10BASE-T data packets are transmitted as 2.5 Vpk Manchester encoded data at 10 Mb/s. 10BASE-T Link pulses are transmitted between data packets to ensure link integrity to the receiving station.

For 100BASE-TX transmission, which borrows from the ANSI X3T12 FDDI TP-PMD specification, packets are imbedded in a continuously scrambled 1 Vpk MLT-3 encoded datastream at an effective data rate of 100 Mb/s. Link integrity for 100BASE-TX is monitored via a Signal Detect function in conjunction with the synchronization status of the receive descrambler.

The majority of CSMA/CD operations remain unchanged between 10BASE-T and 100BASE-TX. Refer to the appropriate IEEE 802.3 specifications for further information.

A detailed review of this design is divided between the repeater's receive and transmit operations. Receive operations include all signaling from the RJ45 media connector to the DP83850 RIC device. Transmit operations include all signaling from the DP83850 RIC to the RJ45 media connector. Sections covering clock distribution and system layout issues are also included.

The 100BASE-TX transmit and receive operations are divided into four basic categories:

- **Physical Medium Dependent (PMD)—DP83223**  
The PMD sublayer section consists of the RJ45-8 media connector, an isolation transformer (magnetics), the DP83223 TWISTER™ transceiver, and the associated interconnections.
- **Physical Layer (PHY)—DP83840**  
The Physical Layer section consists of the DP83840 100BASE-X PHY device and associated support components.
- **Medium Independent Interface (MII)**  
The Medium Independent Interface section consists of receive signal timing parameters and special considerations.
- **Repeater Controller—DP83850**  
The Repeater Controller section describes the fundamental operations of the DP83850 100RIC device.

### 3.0 PMD

This section describes the receive and transmit signal paths between the twisted pair cable and the DP83840 Physical Layer device. The PMD design suggestion given in *Figure 2* should be implemented for each port in a multi-port 100BASE-TX repeater design.

The schematic in *Figure 2* provides electrical interconnection detail for a 100BASE-TX Physical Medium Dependent

(PMD) circuit based primarily on the DP83223 TWISTER. Contact National Semiconductor for information regarding magnetics recommendations.

#### 3.1 PMD Receive

The receive datastream is coupled from the twisted pair cable to the 100BASE-TX repeater port via pins 1 and 2 of the RJ45-8 media connector. The datastream is AC coupled from the RJ45-8 to the DP83223 twisted pair receiver by an isolation transformer. The DP83223 then equalizes the receive signal to compensate for signal degradation caused by the non-ideal transmission line properties of the twisted pair cable. The DP83223 then translates the equalized receive bit stream from MLT-3 to binary and outputs it to the Physical Layer as a Pseudo-ECL (PECL) logic level signal.

Additionally, upon reception of an appropriate input signal, the DP83223 generates a Signal Detect signal which indicates to the DP83840 Physical Layer device that a potentially valid 100BASE-TX signal is present.

#### 3.2 PMD Transmit

The transmit datastream, as sourced by the DP83840 100BASE-X Physical Layer device, is a scrambled NRZI differential PECL signal that is directly connected to the DP83223 twisted pair transceiver device. The DP83223 translates the PECL signal into an MLT-3 encoded signal which is output to the magnetics as a current sourced differential datastream. The signal is AC coupled via the magnetics, to pins 3 and 6 of the RJ45-8 media connector where it is finally coupled to the twisted pair cable.

#### 3.3 PMD Magnetics

The Magnetics Detail diagram given in *Figure 3* provides the required interconnection for the magnetics module within the PMD circuit. Magnetics modules suitable for use with the design outlined herein are available from Pulse Engineering (part# PE68515) and Valor Electronics (part# PT4171). National Semiconductor continues to qualify additional magnetics from various manufacturers. Please contact National Semiconductor for a current list of magnetics qualified for this design.

The schematic given in *Figure 2* does not include the common magnetics circuit as specified by National Semiconductor. This circuit is required to allow full support of Auto-Negotiation. Contact National Semiconductor for further information regarding the common magnetics application and NWay Auto-Negotiation.

The common mode termination circuit included in *Figure 3* suggests one method for decreasing impulse noise sensitivities as well as helping to control EMI radiated emissions.

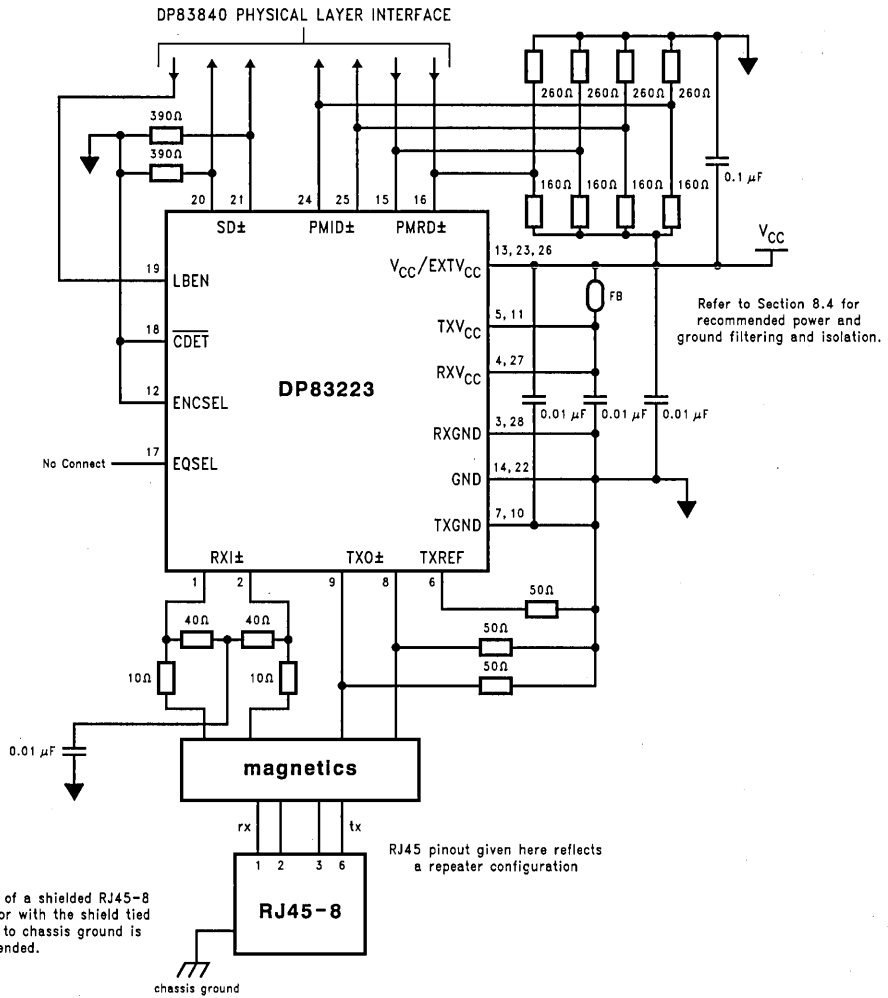
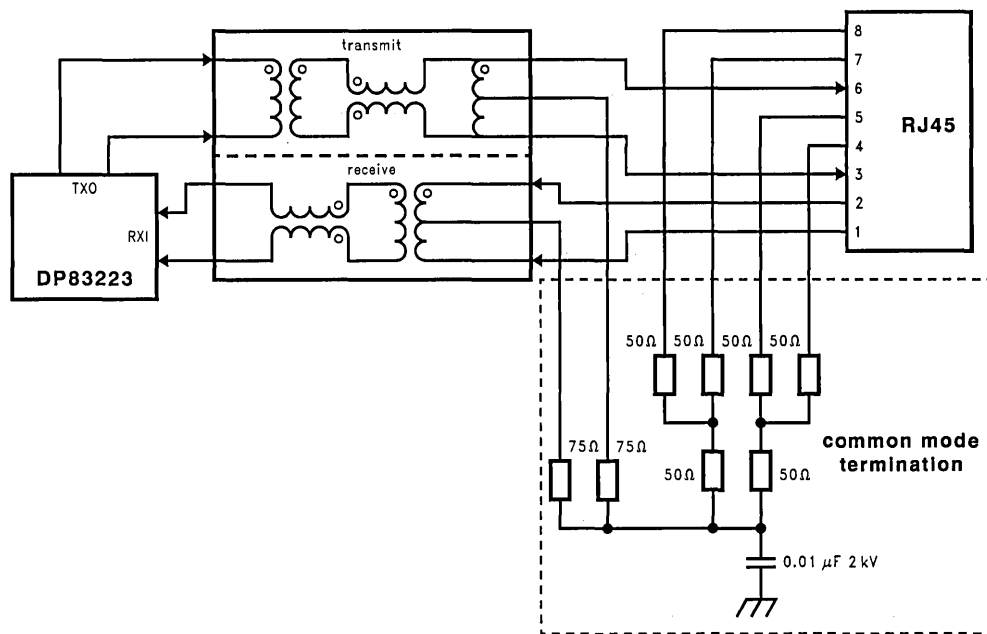


FIGURE 2. 100BASE-TX PMD

TL/F12505-2



TL/F/12505-3

FIGURE 3. PDM Magnetics

#### 4.0 PHY

The DP83840 100BASE-X Physical Layer device incorporates many of the functions required for compliant 100BASE-TX signaling. A connection diagram for the DP83840 is provided in *Figure 4*.

#### 4.1 PHY Receive

The flow diagram in *Figure 5* illustrates the functional blocks within the DP83840 100BASE-X receive channel.

The Physical Layer receive operation begins at the RD $\pm$  and SD $\pm$  inputs of the DP83840. These scrambled PECL signals, as generated by the DP83223 twisted pair receive circuit, are first routed to the integrated 125 MHz clock recovery module which extracts the receive system clock from the asynchronous receive datastream.

The receive data is subsequently converted from serial to 5-bit parallel and routed through the NRZI/NRZ decoder, the descrambler, symbol alignment, and finally the 4B/5B decoder. The nibble wide data is then routed to the MII receive bus outputs RXD[3:0] of the DP83840 where it is accompanied by a synchronous 25 MHz RX\_CLK. Although all of these receive functions can be bypassed, this datastream "conditioning" is required because the DP83850 Repeater Controller IC employs a nibble wide interface.

#### 4.2 PHY Transmit

The flow diagram in *Figure 6* illustrates the functional blocks within the DP83840 100BASE-X transmit channel.

The Physical Layer transmit operation begins at the transmit MII inputs TXD[3:0] of the DP83840. This nibble wide data,

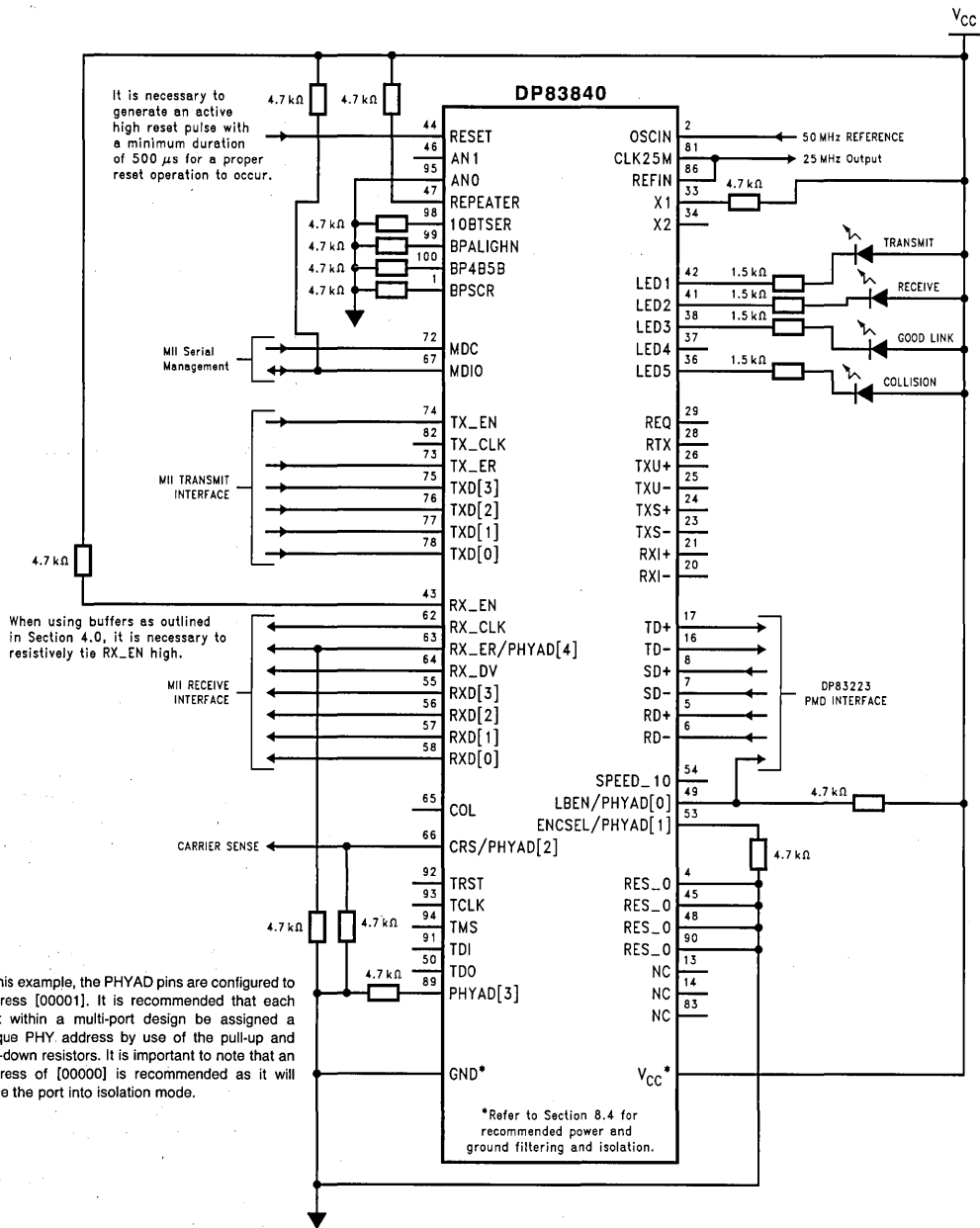
as sourced by the DP83850 Repeater Controller, is first block encoded to 5B symbol wide data, scrambled, and finally post encoded to NRZI format where it is then serialized and routed, MSB first, to the TD $\pm$  outputs of the DP83840. All of these functions are synchronous relative to the internal clock generation module. The 25 MHz reference is used for all of the parallel data functions and a 125 MHz clock is used to serialize and clock out the scrambled datastream.

#### 4.3 PHY Addressing and Serial Management

An unmanaged repeater does not normally require that each of its ports be individually addressed. However, the unique properties of the DP83840 device allow the system designer to benefit from PHY address assignment.

The DP83840 is designed such that the starting value of the transmit scrambler is dependent on the PHY address assigned. This ensures that any number of uniquely addressed DP83840 devices will not be transmitting identical datastreams simultaneously during a transmit operation. Each PHY will be transmitting the same encoded data but at a different scrambled state. This will help to reduce potential EMI radiation problems that may have otherwise resulted from as many as eleven ports simultaneously transmitting the exact same scrambled datastream.

The other benefit to including unique PHY addressing within the 100BASE-TX repeater design is to allow for easy conversion of an unmanaged design to a managed design. While this application note does not focus on the aspects of a managed repeater, a basic understanding of the MII serial management is provided.



TL/F/12505-4

FIGURE 4. DP83840 Connection Diagram

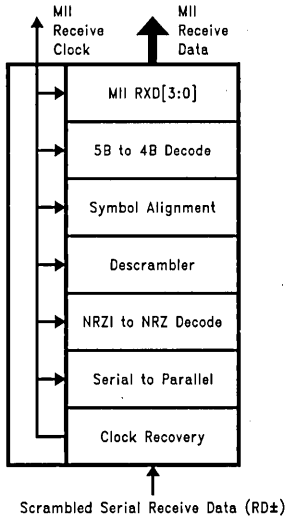


FIGURE 5. PHY Receive Channel

TL/F/12505-5

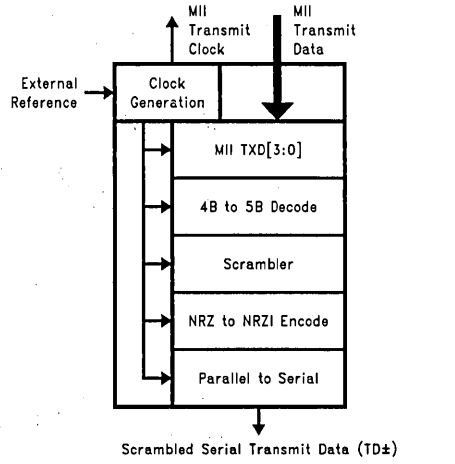


FIGURE 6. PHY Transmit Channel

TL/F/12505-6

PHY address assignment is necessary in a managed multi-port repeater to allow the managing agent to uniquely identify any given port. Given an address, the managing agent can perform read and write operations through the serial access port as defined in Chapter 22 of the IEEE 802.3 $\mu$ /D5 100BASE-T document. This access allows for both statistical gathering and port configuration through read and write operations to the PHY registers.

The address of the DP83840 device is set upon power up or reset and is defined by the pull-up or pull-down state at each of the PHYAD pins, PHYAD [4:0]. Assigning a PHY address of [00000] for any given PHY is not recommended as this will force the port into PHY isolation mode which will disable all transmit and receive activity at that port. For a 12-port repeater, assigning PHY addresses of one through twelve ([00001] through [01100] binary) is recommended.

**4.4 PHY Clocking**

The DP83840 is capable of operation at either 10 Mb/s (10BASE-T) or 100 Mb/s (100BASE-X). Because of this flexibility, a variety of clocking options exist for the DP83840. In the case of a 100BASE-TX application that does not support NWay Auto-Negotiation, the only external clock required is a 25 MHz reference. The DP83840 uses this reference to generate phase locked 25 MHz and 125 MHz to provide the clocks necessary for the 100BASE-X transmit functionality.

Clock reference generation and distribution are covered in Section 7.0

**5.0 MII**

The Medium Independent Interface consists of three basic components. The Serial Management operation (which was briefly covered in Section 4.3), the receive operation

and the transmit operation. Both the receive and transmit interfaces are based on a nibble wide data bus running at 25 MHz allowing a transfer rate of 100 Mb/s.

While the fundamental aspects of MII operation are included herein, a detailed operational description can be found in Chapter 22 of the latest IEEE 802.3 $\mu$  specification for 100BASE-T Ethernet protocols.

**5.1 MII Receive Operation**

The Medium Independent Interface receive operation is a synchronous nibble wide data transfer from the DP83840 Physical Layer to the DP83850 Repeater Controller. This data transfer is initiated when the DP83840 asserts its CRS (Carrier Sense) output which indicates that data reception is in progress. With the REPEATER pin tied high, as shown in Figure 4, the DP83840 will only assert CRS during a receive event to ensure proper interoperability with the DP83850 device. The CRS output of the DP83840 is asynchronous to the RX\_CLK output.

Upon reception of the CRS signal from an active port, the DP83850 will assert the corresponding RX\_EN (Receive Enable) signal. This enables the RX\_CLK, RX\_ER, RX\_DV, and RXD[3:0] outputs of the DP83840 to become active. These outputs are normally TRI-STATE when the RX\_EN input is not asserted.

The relative timing of the MII receive signals is dependent on the quality of the receive operation. There are primarily three different receive scenarios which represent:

- Reception without error(s)
- Reception with error(s)
- False carrier indication

These scenarios are illustrated by the timing diagrams in Figures 7, 8 and 9 respectively.

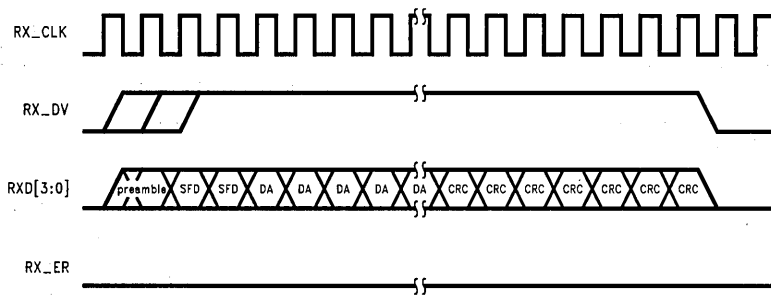


FIGURE 7. MII Reception without Error(s)

TL/F/12505-7

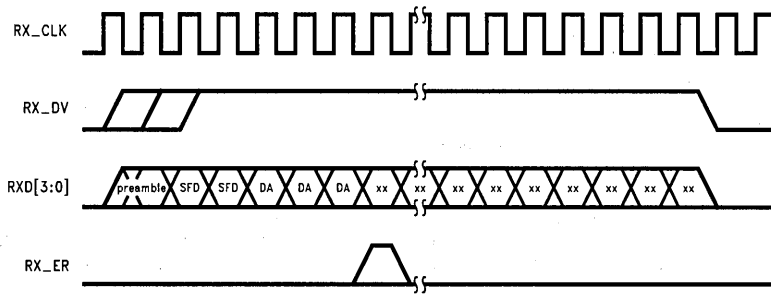


FIGURE 8. MII Reception with Error(s)

TL/F/12505-8

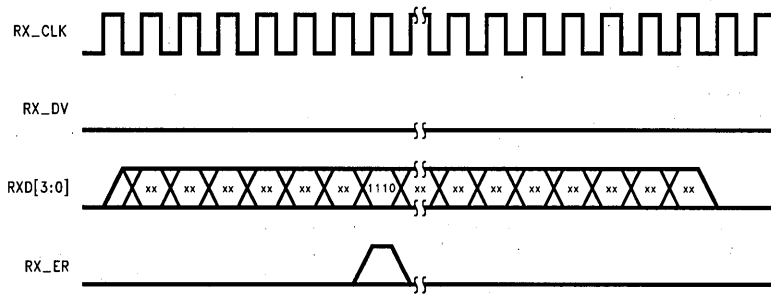


FIGURE 9. False Carrier Indication

TL/F/12505-9

The relative timing for "reception without error(s)" illustrates the desired sequence of events which occur during reception. Upon the reception of a packet, the DP83840 asserts its CRS output signal. The DP83850 then responds by asserting its RX\_EN output signal. Upon reception of the RX\_EN signal, the DP83840 activates each of its MII receive outputs. RX\_CLK will first begin to run at 25 MHz. RX\_CLK will then begin clocking out the nibble wide data RXD[3:0]. Coincident with the preamble on RXD[3:0], RX\_DV (data valid) will assert and remain so for the duration of the receive packet transmission to the DP83850. RX\_DV will deassert immediately following the final data nibble. RX\_ER remains low for the duration of the packet indicating an error free packet.

The relative timing for "reception with error(s)" is similar to the previous case except that the RX\_ER signal is asserted during the packet reception operation. This indicates that the DP83840 Physical Layer device has detected some form of data error during reception. Errors resulting from improper frame alignment can cause assertion of RX\_ER.

The third scenario, "false carrier indication", is an indication that the DP83840 detected invalid data code groups which proceeded the starting delimiter of the packet. Additionally, "false carrier indication" will occur upon detection of an invalid stream termination sequence.

For further detail regarding these MII data reception scenarios, refer to the latest version of the IEEE 802.3 $\mu$  specification for 100 Mb/s Ethernet protocols.

## 5.2 MII Receive Physical Connection

The DP83850 Repeater Controller device is capable of supporting up to twelve DP83840 Physical Layer devices in conjunction with as many DP83223 transceiver devices. In a fully loaded architecture, where all twelve ports are designed in, there are special considerations regarding the physical layout of the MII receive interface.

### 5.2.1 MII Receive Bus Considerations

Due to the nature of the Ethernet Repeater architecture, where one repeater controller receives data from as many as twelve separate Physical Layers, each of the Physical Layer devices must share a common receive data bus. This bus terminates at the receive inputs of the repeater controller.

Specifically, twelve DP83840 devices share a single receive data bus for data transfers to the DP83850 Repeater Controller. The operation of the repeater configuration allows only one DP83840 MII receive output to be active at any given time. All inactive DP83840 devices TRI-STATE their MII receive data outputs. Theoretically, this would allow the single active DP83840 to transfer the receive packet to the DP83850 without interaction with the remaining inactive DP83840 devices. In practice, however, the DP83840 must not only drive the signal traces between itself and the inputs of the DP83850, but also the entire common data bus with all of the inherent distributed capacitance and trace routing aberrations.

The distributed trace capacitance, in addition to the capacitance of the TRI-STATE outputs of the inactive DP83840s can total to 150 pF or higher. This distributed capacitive load can cause waveform anomalies and reduce the signal integrity within the MII receive interface. The RXD[3:0], RX\_DV, and RX\_ER outputs of the DP83840 are all subject to this additional capacitive loading.

### 5.2.2 MII Receive Buffering

To significantly reduce the effects of the distributed capacitive loading, an octal buffer can be placed between each DP83840 and the DP83850 Repeater Controller device. Octal buffers such as the National Semiconductor ABT541 provide significant output current drive capability which effectively improves the signal integrity to help ensure robust MII receive data transfer.

While the DP83840 provides sufficient MII receive clock-to-data setup and hold times for the DP83850, this timing relationship can be optimized by utilizing a latch at the input of the DP83850 to re-time the data relative to the RX\_CLK. Additionally, the use of an inverter for the RX\_CLK will ensure virtually ideal set and hold timing for the DP83850 MII receive inputs. While issues such as repeater cascading (often referred to as stacking) are not specifically covered in this application note, it is important to understand that the inclusion of the latch and inverter as described herein is critical to ensure robust operation of a multi-repeater cascaded system.

Figure 10 provides the suggested interconnection for the receive MII (CRS is also shown) of a twelve port repeater design. This design approach includes recommendations for buffering, latching, and general component placement guidelines to help ensure robust interface performance.

The use of the ABT541 octal buffers requires that an active low  $\overline{OE}$  signal be generated so that the active receive port can source its data to the DP83850. As illustrated in Figure 10, by inverting the RXE bus and connecting the inverted RXE signals to the appropriate ABT541 octal buffers, the active Receive MII data and clock is enabled. Since no more than one ABT541 octal buffer can be asserted at any given time, the remaining disabled ABT541 devices will TRI-STATE which eliminates contention on the shared bus.

The RX\_EN inputs to the DP83840 devices should be tied high such that they are never allowed to TRI-STATE. The ABT541 buffers will perform the necessary bus TRI-STATE in place of the DP83840 devices. There are important design recommendations which, if incorporated, will help to ensure proper interface operation.

When a given design requires that multiple signal sources share a common bus with a single destination, it is difficult to maintain good signal transmission properties. The best way to reduce signal reflection anomalies in such a shared bus configuration is to minimize the signal trace lengths. By placing the ABT541 octal buffers and the ABT574 latch as close as possible to the DP83850, the area of shared signal routing for RXD[3:0], RX\_DV, RX\_ER, and RX\_CLK is minimized. Allowing longer trace lengths from each of the DP83840 PHY devices to their respective ABT541 buffers is acceptable because each PHY now has only a fan-out of one (individual ABT541 input) per signal trace. Since the RXE and CRS signals, between the DP83850 and the DP83840s, are each routed separately (not a shared bus) good signal integrity is easier to achieve but not guaranteed. Noise coupling onto these lines can cause improper operation and must be considered during board layout. As with any relatively high speed design, it is helpful to minimize signal trace lengths and minimize crosstalk wherever possible.

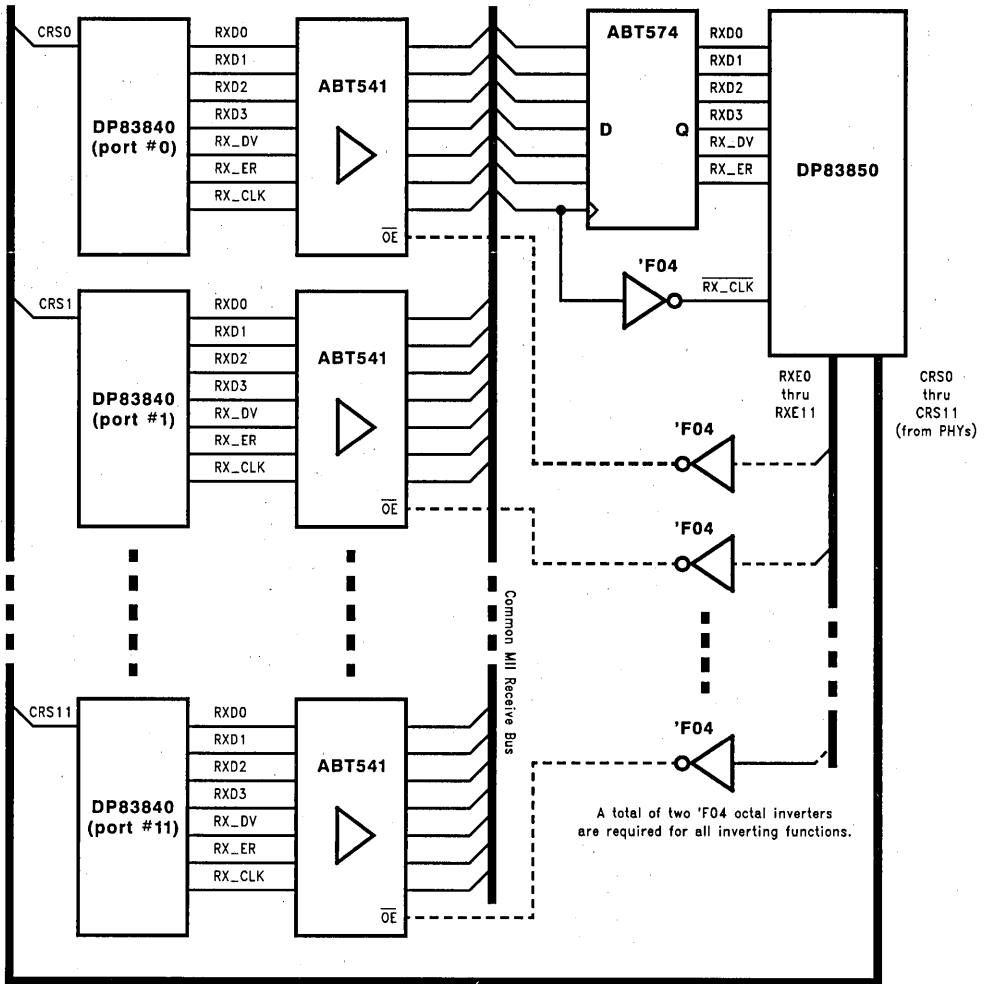


FIGURE 10. MII Receive Connection Recommendation

TL/F/12505-10



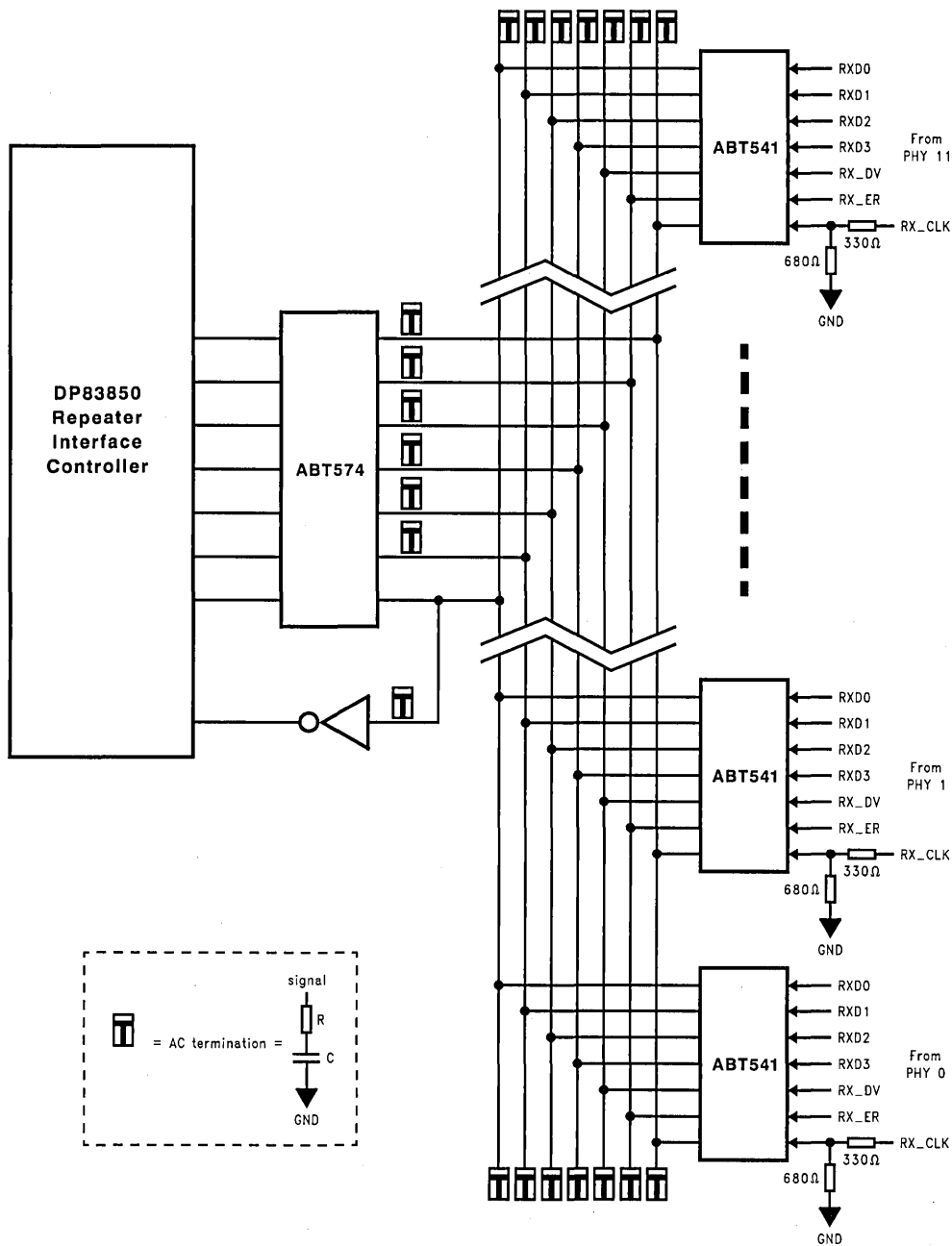


FIGURE 11. MII Receive BUS Termination

TL/F/12505-11

### 5.2.3 MII Receive Bus Terminations

The use of bus terminations may also be implemented in an effort to control signal integrity on the common MII bus. The option for AC terminators at both ends of the bus, as well as at the inputs to the DP83850 device, should be incorporated into the design as illustrated in *Figure 11*. By populating some combination of these AC termination components and by experimenting with the component values, the signal aberrations inherent to a given common bus layout may at least be partially compensated. A standard AC termination consisting of a series resistor and capacitor to ground should be used. Component values of 50 pF and 100Ω are good typical values to begin the empirical experimentation of finding the best possible AC termination values for a given layout design.

Placing AC terminations at each CRS input to the DP83850 as well as at the OE inputs (RXE) of the ABT541 buffers will help to minimize signal aberrations on these traces as well as reduce noise that may couple from these traces onto other signals.

Another concern is the potential for duty cycle distortion of the RX\_CLK signals. This can occur due to the fact that the CMOS level RX\_CLK signal must travel through the ABT541 buffer as well as through the 'F04 inverter before it arrives at the DP83850 input. More specifically, the limited slew rate CMOS output of the DP83840 will be "sampled" at approximately 1.5V by the ABT541 buffer because the buffers input threshold is nominally 1.5V. This can result in duty cycle distortion that will then propagate on through the 'F04 inverter and into the DP83850 device. To minimize the potential for duty cycle distortion, it is recommended that a simple voltage divider be placed at the RX\_CLK output of each PHY device. This voltage divider as depicted in *Figure 11* will provide the ABT541 input with a nominal 3V pk-pk clock signal which will help to minimize the potential for duty cycle distortion. This technique is not necessary at the related outputs of the ABT541 buffer or the 'F04 inverter due to their output drive characteristics.

### 5.3 MII Transmit Operation

The MII transmit operation is a synchronous nibble wide data transfer from the DP83850 RIC device to the DP83840 PHY devices.

Upon reception of CRS from the DP83840 PHY which is actively receiving data from the network, the DP83850 asserts the TXE (transmit enable) lines to all of the idle DP83840 PHYs. Along with the assertion of the TXE signals, the DP83850 also transmits the MII Transmit data consisting of TXD[3:0] and TX\_ER. This data is then repeated out to the network via all of the active transmitting DP83840 devices.

The relative timing of the MII transmit operation is depicted in *Figures 12 and 13*. These timing diagrams illustrate a normal transmit operation without errors and a transmit operation with error propagation.

The relative timing for "transmission without error(s)" in *Figure 12* illustrates the desired sequence of events which occur during MII transmission.

The relative timing for "transmission with error propagation" is similar to the previous case except that the TX\_ER signal is asserted during the packet transmission operation. This ensures that any receive packet error will be repeated in the transmit operation. By repeating the packet error to the destination node, that node can process the error without interdiction from the 100BASE-TX repeater system.

#### 5.3.1 Transmission and Collisions

100BASE-TX repeater applications process collisions differently than 100BASE-TX node applications.

A 100BASE-TX node based on the DP83840 device will generate a CRS signal, which appears on the "CRS" output of the DP83840, for either packet reception or packet transmission. This indicates to the MAC Layer that the Physical Layer is operating properly. However, when CRS is asserted due to simultaneous packet transmission and reception, the DP83840 will assert its "COL" output which indicates a collision condition to the MAC Layer. This causes a JAM pattern to be transmitted and then attempts another transmission per the CSMA/CD back-off algorithm.

In a 100BASE-TX repeater based on DP83850 and DP83840 devices, only packet reception will cause the assertion of the CRS signal. CRS remains unasserted for transmission operations. In this case, simultaneous reception on two or more repeater ports will cause CRS to assert from two or more DP83840 devices. Multiple CRS signals are processed by the DP83850 RIC device and interpreted as a collision event (refer to *Figure 14*). Upon a collision event, the DP83850 will send a JAM pattern to all ports for the remainder of the transmission period.

To select between the node and repeater modes of operation, the DP83840 "REPEATER" input pin should be properly configured. As shown in *Figure 4*, a logic high level on this pin will configure the DP83840 for repeater operation. A logic low on the "REPEATER" input will configure the DP83840 to node mode.

#### 5.3.2 MII Transmit Physical Connection

The physical connection between the DP83850 and the DP83840 required to support an MII transmit operation is illustrated in *Figure 15*. Unlike the MII receive bus, the transmit bus does not require special buffering. Since the signal path of the MII transmit section is essentially fixed, a single termination point at the end of each MII transmit signal is sufficient to provide good signal integrity.

TXD[3:0] and TX\_ER are the only bussed MII transmit signals. Because the DP83850 provides twelve TX\_EN outputs, these signals can be routed individually to each corresponding DP83840 device. The TX\_CLK output of the DP83840 device need not be connected to the DP83850 input because both devices operate synchronously from the same master reference clock which is used for the MII transmit data transfer.

Every effort should be made to minimize the overall length of the MII transmit bus. Additionally, it is good practice to route each MII transmit signal as a single contiguous path from the DP83850 to the final AC termination point as shown in *Figure 15*. Additionally, by placing 22Ω series resistors at the beginning of each TXD signal line, potential signal aberrations can be further controlled.

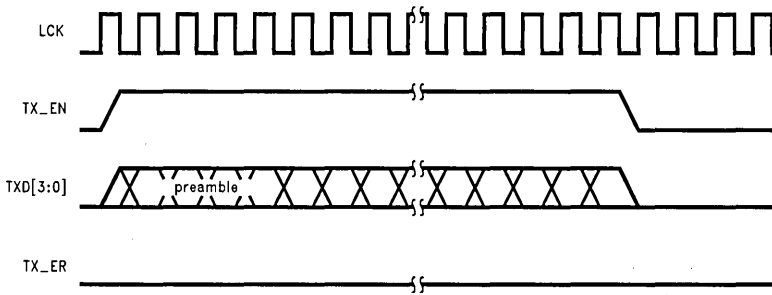


FIGURE 12. MII Transmission without Error(s)

TL/F/12505-12

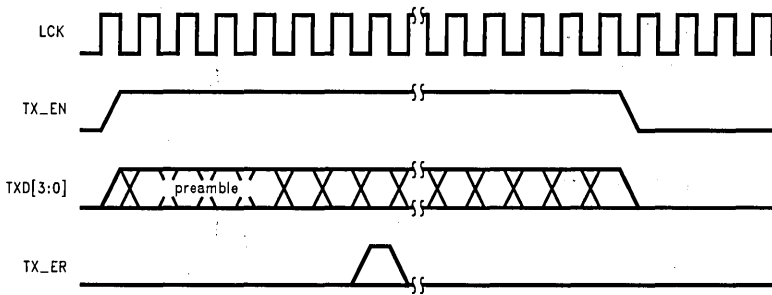


FIGURE 13. MII Transmission with Error Propagation

TL/F/12505-13

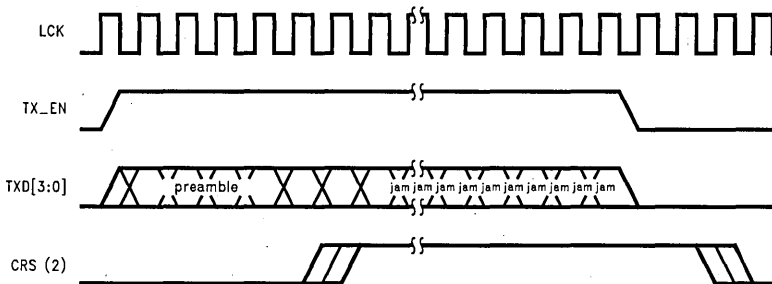


FIGURE 14. MII Transmission with Collision

TL/F/12505-14

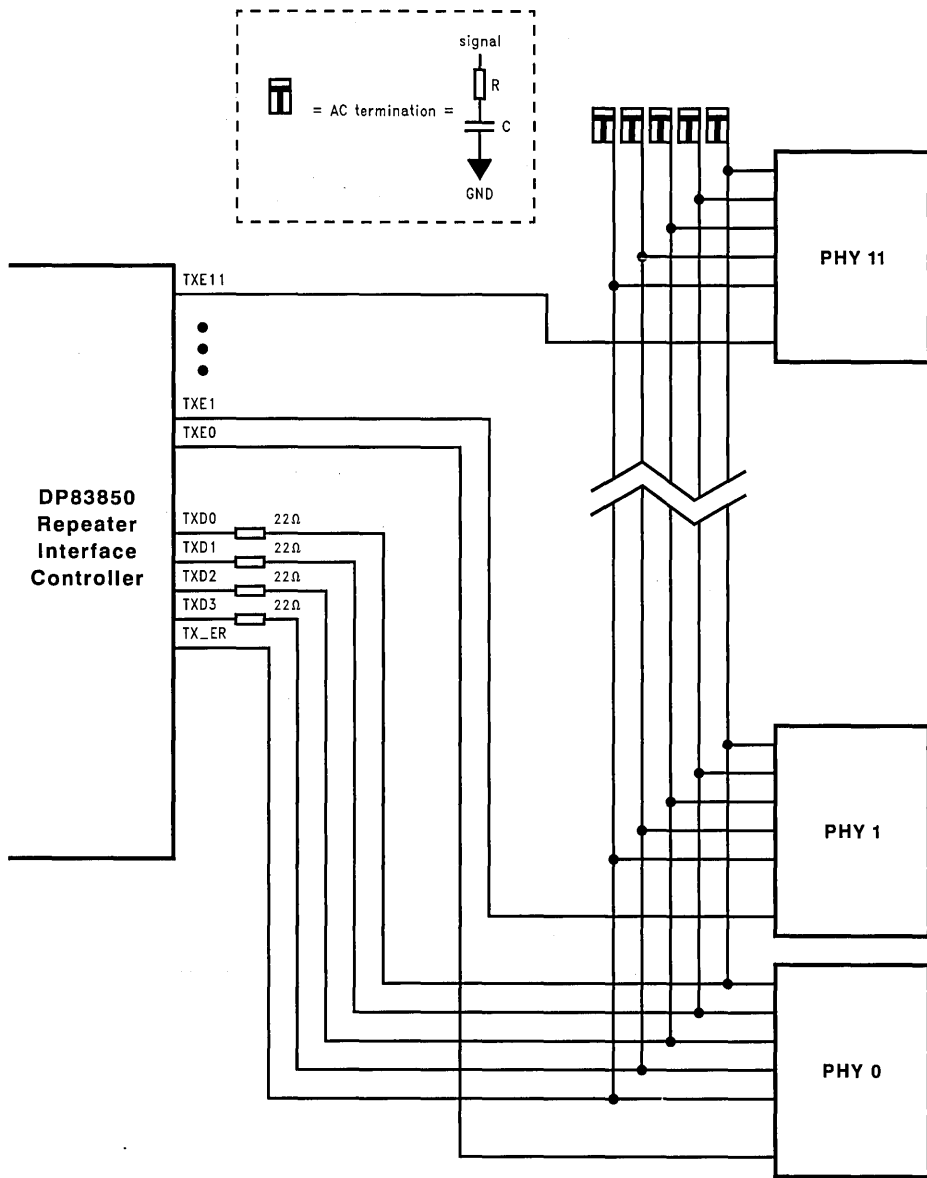


FIGURE 15. MII Transmit BUS Physical Connection and AC Termination

TL/F/12505-15

## 6.0 100RIC REPEATER CONTROLLER

The DP83850 100RIC Repeater Controller IC provides the basic packet control for the 100BASE-TX repeater system. This controller includes a significant feature set which allows it to provide a number of different functions. For this 12-port unmanaged application, however, only those functions that are required for unmanaged operations are considered. Features relating to the Inter-RIC™ bus as well as connectivity to a Media Access Controller (MAC) of a Repeater Information Base (RIB) are outside the scope of this document. *Figure 16* provides the connection diagram for the DP83850.

The straightforward interconnect of the DP83850 to the MII results in a simple overall implementation. The only special connection considerations relative to the DP83850 pertain to the `IR_VECT[4:0]`, `~IR_COL_OUT` and `~IR_COL_IN` pins as well as the `~IR_ACTIVE` and `~ACTIVE0` pins. These Open-Collector pins each require a 1 k $\Omega$  pull-up resistor to  $V_{CC}$  and should be connected as indicated in *Figure 16* to ensure proper operation in an unmanaged application. It is also important to connect `~IR_COL_OUT` directly back to `~IR_COL_IN` to ensure proper operation in an unmanaged application. Each Open Collector pin of the DP83850 is indicated as so with an "(OC)" as shown in *Figure 16*.

Those pins or groups of pins marked with the reference "(MAN)" are intended to support managed applications and the Inter-RIC bus. It is recommended that these pins be connected to a header strip or similar connector in order to provide an easy migration from an unmanaged to a managed repeater design. Please refer to the DP83850 data-sheet for detailed information regarding a managed application.

## 7.0 REFERENCE CLOCK

The DP83840 and DP83850 devices require a reference clock in order to operate. This reference clock is used for all of the synchronous transmit operations of the 100BASE-TX repeater. While there are several methods for implementing the reference clock in a 100BASE-TX repeater application, the following methods provide robust and proven solutions for designs that support Auto-Negotiation as well as those that do not.

### 7.1 Reference Clock Option One

*Figure 17* illustrates the circuitry and interconnection recommended for generating and distributing the system reference clock for an unmanaged 100BASE-TX repeater that does not support Auto-Negotiation.

A properly filtered 50 ppm 25 MHz oscillator will provide a stable source for the reference clock. The output of this oscillator should be routed first to the 74CT2525 and from there, to the DP83850 input. By including an option for AC termination at the input to the DP83850, potential signal reflections can be partially controlled.

In order to distribute the clock to multiple PHY devices, a low skew clock driver such as the 74CT2525 provides a simple method for deriving multiple synchronous phase aligned clocks with sufficient signal drive. With the proper clock signal routing, a single output from the 74CT2525 can effectively source two PHY devices. Additionally, the inclusion of an R/C network, placed as close as possible to each of the 74CT2525 outputs, will help to control the high frequency components of the reference clocks for EMI control purposes. As illustrated in *Figure 17*, by routing the clocks to the PHY devices in a daisy chain configuration and providing AC termination at the end of the signal paths, signal integrity can be maintained. It should be noted that for each PHY device the 25 MHz clock should be routed to both the REFIN and OSCIN inputs.

Supplying the clock to the REFIN input provides the PHY with the required reference. Supplying the clock to the OSCIN input is necessary to ensure proper initialization of the PHY device. While the OSCIN input normally requires a 50 MHz reference, a 25 MHz reference may be used when Auto-Negotiation is not included in the design.

### 7.2 Reference Clock Option Two

*Figure 18* provides a recommended reference clock scheme for a 100BASE-TX repeater that does support Auto-Negotiation.

This clock generation and distribution scheme is similar to that given in Section 7.1 except that the PHY devices, in order to support Auto-Negotiation, must be sourced 20 MHz along with the required 25 MHz clock. Therefore, the second oscillator is required to generate the 20 MHz clock signal. While it would be possible to simply source a 50 MHz clock to each PHY for both 100 Mb/s and Auto-Negotiation functionality, the use of 25 MHz and 20 MHz clocks will keep distributed system clock signals below the 30 MHz limit imposed by the FCC for EMI purposes.

The R/C networks placed at each output of the 74F2525 must also be considered. In order to avoid too much high frequency attenuation the values for the resistors and capacitors must be selected to provide optimum frequency control for the 25 MHz signals being distributed. Similarly the AC terminations located at the end of each routed clock signal must also be selected to provide optimal termination relative to the 25 MHz clock signal.

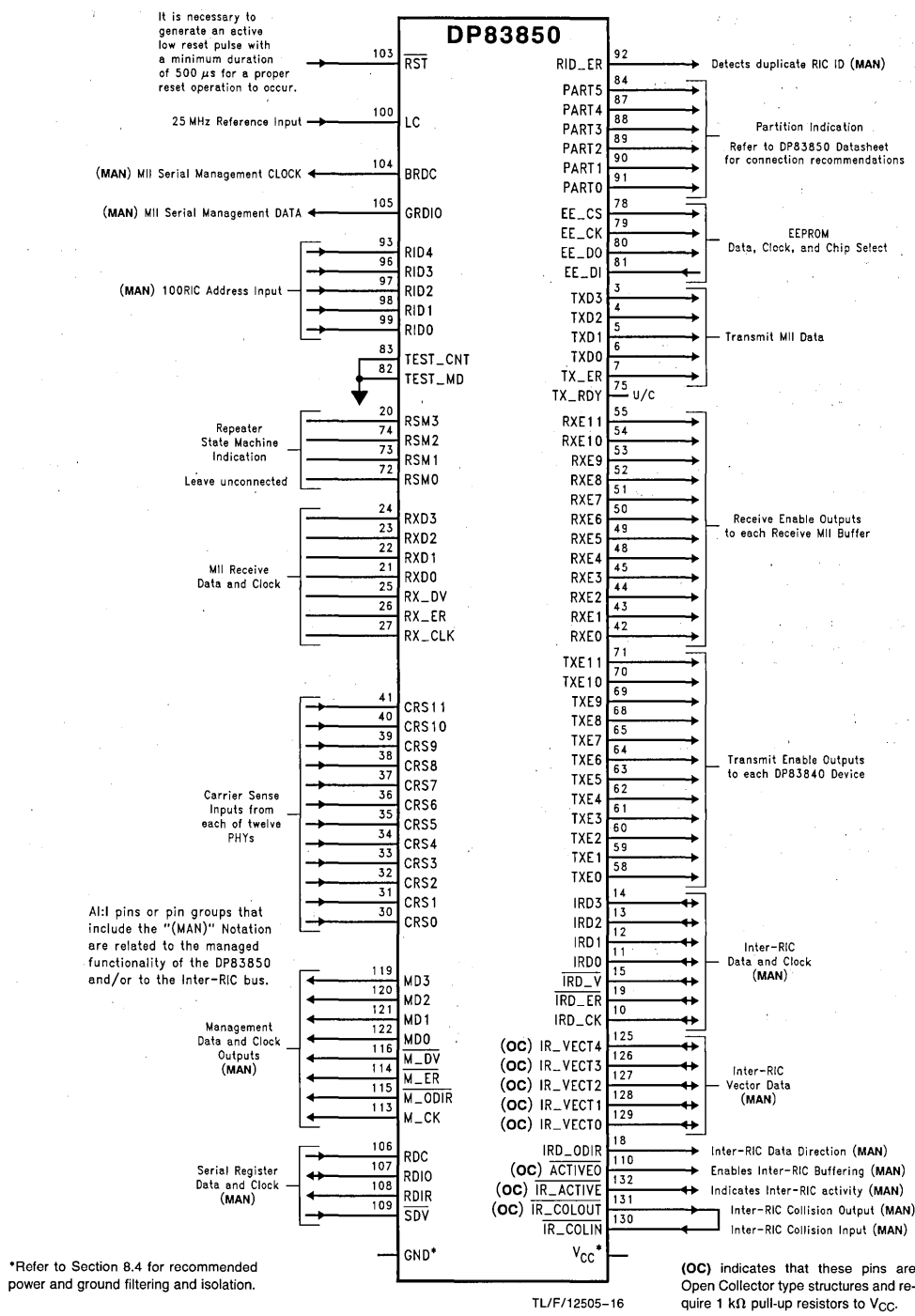


FIGURE 16. DP83850 Connection Diagram

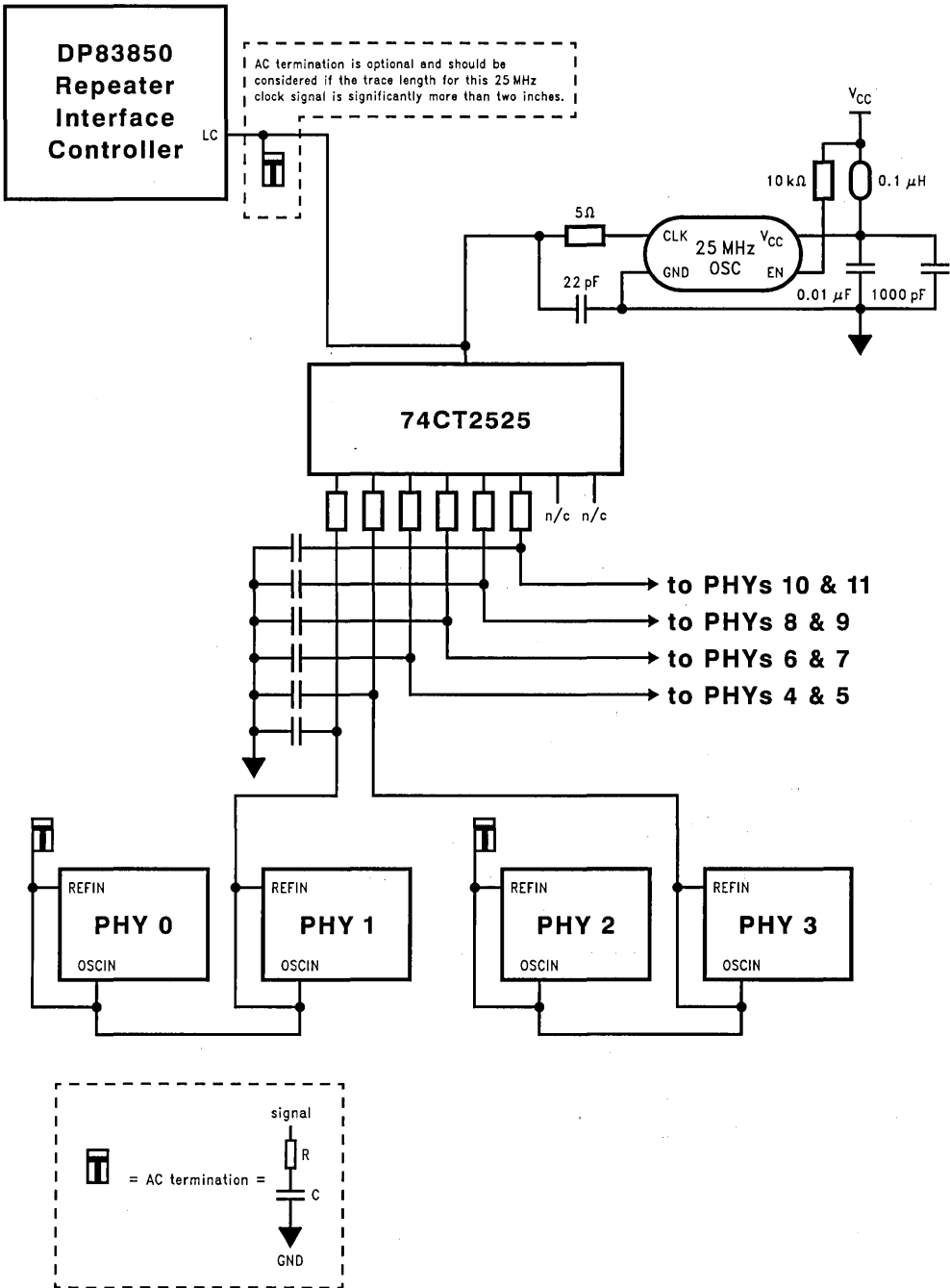


FIGURE 17. Reference Clock Recommendation for a Non-Auto-Negotiating 100BASE-TX Repeater

TL/F/12505-17

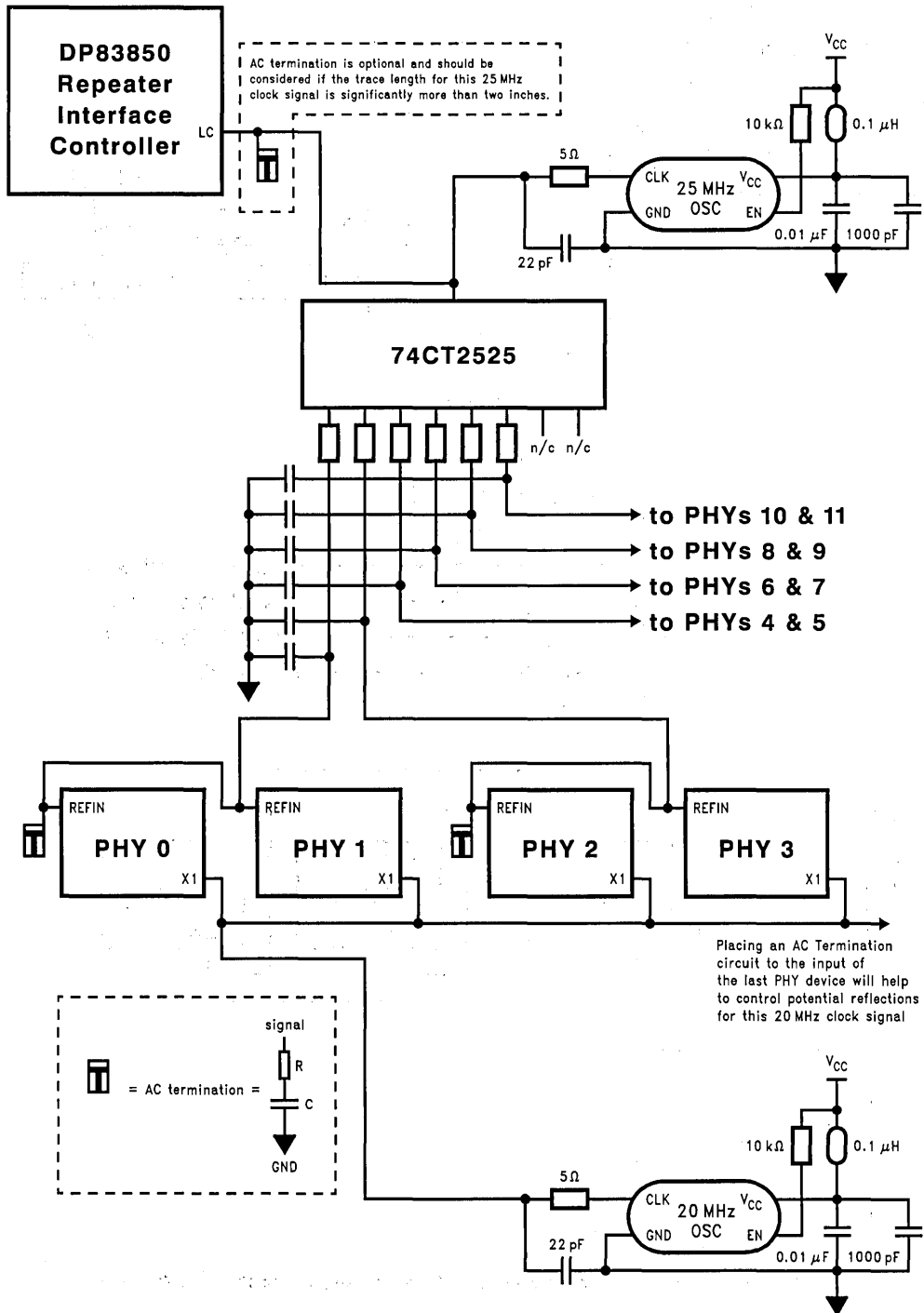


FIGURE 18. Reference Clock Recommendation for An Auto-Negotiating 100BASE-TX Repeater

TL/F/12505-18



## 8.0 PHYSICAL LAYOUT

The goal of any complex system design, especially one which includes both analog and digital functionality, is to achieve the most robust system performance possible. Performance aspects such as fault tolerance, bit-error-rate, EMI, synchronous timing, and general signal integrity must all be considered.

The right combination of component placement, signal routing practices, and power supply distribution will yield a robust and reliable system.

This section explores the physical design aspects that must be considered when designing an unmanaged 100BASE-TX repeater. The latest National Semiconductor 100k ECL Databook and Design Guide provides considerable detail regarding the theory and practice of system design. With focus on such areas as "Transmission Line Concepts" and "Power Distribution and Thermal Considerations", the EGL Design Guide provides substantial insight to many physical layout parameters and their subsequent effects on signal integrity.

### 8.1 Component Placement

The relative placement of the individual active and passive components within a 100BASE-TX repeater is essentially defined by some important design considerations: cost, board area, and performance.

*Figure 19* illustrates one potential component layout approach that will yield good signal integrity and good overall performance. The intent of this layout is to minimize the required board area while still optimizing the relative component placement. The number of layers required to support this design may vary depending on the signal routing density. In general however, a six-layer board comprised of one ground, one supply, and four signal layers should suffice.

The layout of the dynamic transmit and receive signals at the twisted pair transceiver interface and the MII is critical. Optimizing the relative orientation of each DP83840 with its corresponding DP83223 helps to minimize the signal routing required for the 125 Mb/s serial data as illustrated in the

detail diagram given in *Figure 19*. By placing the DP83850 RIC device near the first port of the repeater, the MII transmit signals can be routed easily as described in Section 5.3.2. Additionally, by clustering the ABT541 buffers close to the DP83850, the common MII receive bus size is minimized as described in Section 5.2.2

While relative component placement is important, board area constraints pose certain limitations. If the component placement suggestion given in *Figure 19* is implemented, there would be very little area remaining on the top side of the system board for the required passive components. By placing a majority of the passive components (resistors, capacitors, and ferrite beads) on the bottom side of the board, the system layout constraints can still be met.

### 8.2 Signal Routing

This section focuses on several aspects of signal routing which can contribute to robust signaling within the 100BASE-TX system design. *Figure 20* illustrates one signal routing example for the critical 125 Mb/s sections of the design. Considerations such as controlled impedance trace routing and termination techniques are important. Recommendations for both the high speed (125 Mb/s) and the lower speed (25 Mb/s) signals are provided.

#### 8.2.1 Controlled Impedance of Signal Traces

It is important to incorporate controlled impedance routing for those signal traces which carry the 125 Mb/s serial data. Either standard micro-strip or strip-line techniques are recommended. Please refer to the latest F100K ECL Design Guide from National Semiconductor for detailed information regarding transmission line concepts.

It is important to choose an impedance of 50 $\Omega$  for each trace that carries 125 Mb/s data between the RJ-45, the magnetics, and the DP83223 transceiver. This is necessary in order to match the 100 $\Omega$  differential impedance of the unshielded twisted pair cable.

The 125 Mb/s PECL signals that connect between the DP83223 and the DP83840 can be routed as 100 $\Omega$  impedance traces and terminated with 100 $\Omega$  terminations to help reduce system power requirements.

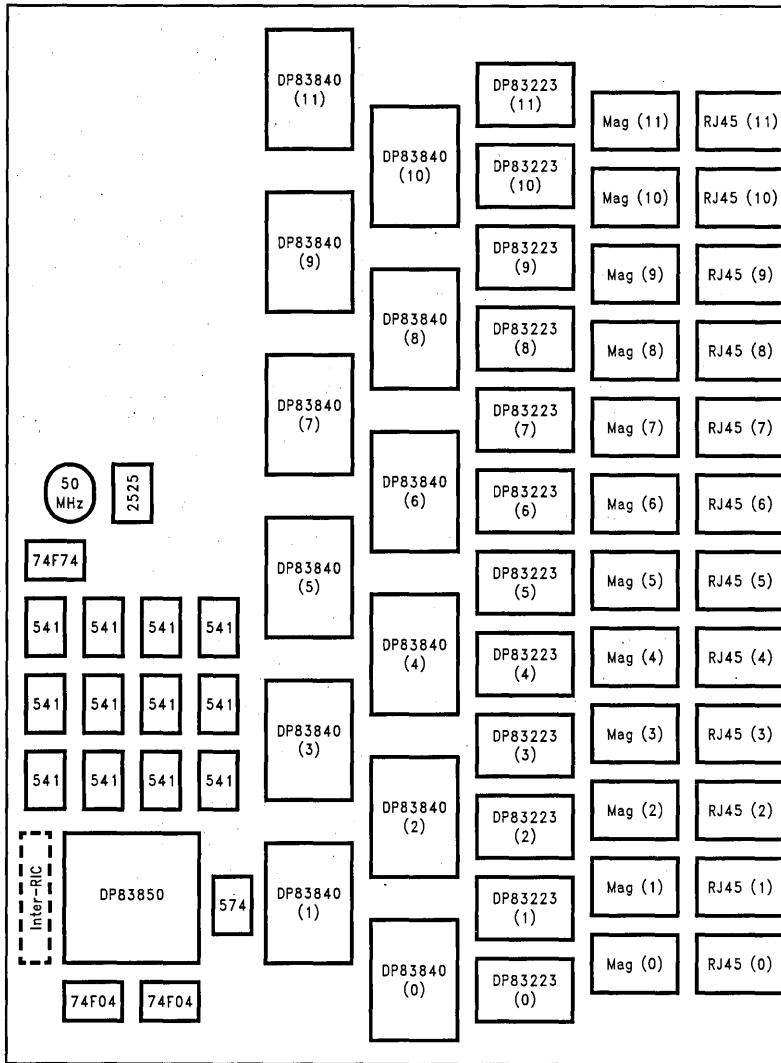
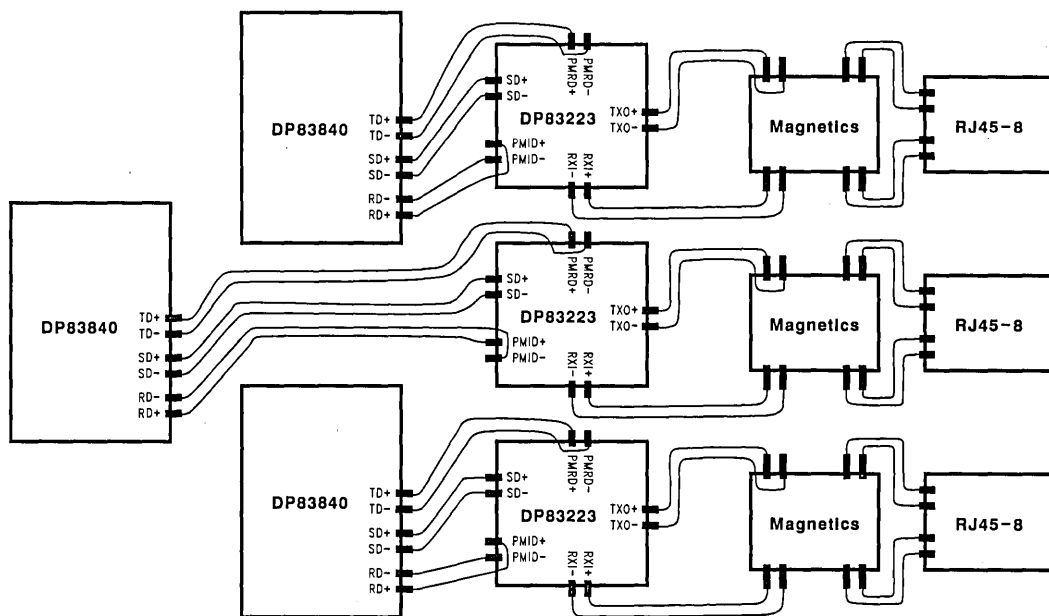


FIGURE 19. Component Placement for a 12-Port 100BASE-TX Repeater

TL/F/12505-19



TL/F/12505-20

FIGURE 20. 125 Mb/s Signal Routing Example for Non-Auto-Negotiating 100BASE-TX Repeater

### 8.2.2 Signal Trace Termination

Proper termination of a high speed signal trace is essential in order to maintain an effective transmission line. In general, it is practical to regard any high speed (125 Mb/s) signal trace that is longer than one inch in total distance as a transmission line. High speed (125 Mb/s) signal traces less than one inch may still provide robust signal transfer without employing some of the transmission line design techniques. Additionally, lower speed signal traces (i.e. 25 Mb/s), do not usually require the added component cost and power requirements of controlled transmission lines. *Figures 21 and 22* provide two alternatives for Pseudo ECL Termination techniques. Either of which will provide good signal integrity for the 125 Mb/s signals routed between the DP83840 and the DP83223 devices.

### 8.2.3 General Guidelines

General guidelines regarding optimal signal trace routing practices include:

- Minimal length controlled impedance signal traces to minimize reflections and decrease noise sensitivities
- Matched length differential signal traces to minimize jitter
- Radiused routed trace corners of  $>45$  degrees
- Minimized number of vias for any one given signal trace
- ECL terminations placed close to signal destination
- All controlled impedance traces routed directly over or under uninterrupted power or ground planes on adjacent layer(s)

### 8.3 Power Supply Recommendations

Careful power supply filtering and isolation practices can provide a minimized noise environment for each of the unique digital and analog sections of both the DP83223 and DP83840 devices. Additionally, providing substantial planes for both the system and chassis grounds will help to minimize potential noise sources that may detract from good EMI performance.

#### 8.3.1 $V_{CC}$ Plane Partitioning

*Figure 24* illustrates the recommended  $V_{CC}$  plane partitioning. In this example,  $V_{CC}$  is separated into three distinct sources: Digital, PECL, and Analog. For purposes of reference, the Digital section is referred to as a plane whereas the PECL and Analog sections are referred to as islands.

The Digital  $V_{CC}$  plane supplies power to the main system which includes most of the digital devices such as the DP83850, buffers, inverters, and other potential devices. Additionally, the Digital  $V_{CC}$  also supplies power to a majority of each DP83840 device. Refer to Section 8.4 for detailed power supply connection information.

The PECL (Pseudo ECL)  $V_{CC}$  is divided up into one island per port. These islands supply power to the Pseudo ECL interface between each DP83223 and its corresponding DP83840. In this example, each port is given its own PECL island in order to minimize noise coupling between adjacent ports.

The Analog  $V_{CC}$  is also divided up into one island per port. These islands supply power to the sensitive analog Transmit and Receive sections of the twisted pair Transceivers. It is important to keep the power sources for these analog sections isolated from the Digital and PECL  $V_{CC}$  planes and islands in order to provide as much noise margin as possible. This helps to ensure robust signaling in extreme cases such as severely attenuated receive signals caused by very long (100 Meters) cable lengths between connections.

The separation between each  $V_{CC}$  plane and/or island on a single board layer should be at least 0.05 inches. This will help to reduce capacitive coupling which may occur as a fringe phenomenon at the edge of each segmented plane or island. Additionally, all island or plane corners  $>45$  degrees should be radiused.

#### 8.3.2 Ground Plane Partitioning

The ground plane should not be partitioned into separate islands. Partitioning of the ground plane can lead to increased EMI emissions which may make the system non-compliant to specific FCC regulations. *Figure 25* provides the recommended ground layout and specifies the division between and placement of the system ground versus the chassis ground.

By keeping the chassis ground back from the edge of the system motherboard by approximately 0.25 inches and simply voiding that gap of any copper will help to reduce any potential fringe radiation that may occur during system operation. This is permissible as no active traces need to be routed in this area anyway.

#### 8.3.3 Board Layers

*Figure 23* illustrates one potential option for board layer assignment. This option places particular emphasis on EMI concerns. The eight layer approach is recommended in order to accommodate the significant number of trace interconnections while still allowing sufficient area dedicated to power and ground planes and islands.

As denoted, Layer One (top layer) is normally the component side and this is where all or most of the active devices should be located. By minimizing the amount of trace routing on this layer and maximizing the system ground area, partial EMI shielding can be achieved. It is also important to route the Chassis ground, as shown in *Figure 25*, on this top Layer.

Layer Two should be dedicated to full system ground and chassis ground as illustrated in *Figure 25*.

Layer Three may be used as a signal routing layer. It is important to attempt to route any one continuous trace, or differential trace pair, on a single board layer. As an example, the designer may decide to route all or most of the 125 Mb/s traces on this layer and devote another layer to the slower MII signals.

Layer Four may be used as the  $V_{CC}$  layer where the Digital  $V_{CC}$  plane and each of the carefully designed  $V_{CC}$  islands, as depicted in *Figure 24*, will reside.

The next layer, Layer Five, may also accommodate signal routing.

Layer Six should be dedicated to full system ground and chassis ground as illustrated in *Figure 25*.

Layer Seven may also accommodate signal routing.

Layer Eight should be physically similar to Layer One. This is traditionally known as the solder-side of the board, however, this side will also carry many passive components. Therefore, the bottom layer doubles as a component side and solder side. As with Layer One, all undedicated space on this layer should be filled in with copper and tied to system ground, except where Chassis ground is located as illustrated by *Figure 25*. Signal routing should be minimized on this layer wherever possible.

By placing System and Chassis ground on both the top and bottom layers, a virtual Faraday cage is produced which will aid in the control of EMI emissions. If more than eight layers are required to accommodate the signal routing, it is

important to interleave these additional signal layers with ground or power planes in order to increase shielding and to achieve good trace impedance control.

Embedding the signal layers within the board does increase the number of vias which will be required to route to and from the active and passive components on the top and bottom layers. This will add some cost to the board and can result in transmission line traces that are marginally less than ideal. However, the potential for good EMI performance is worth the compromise.

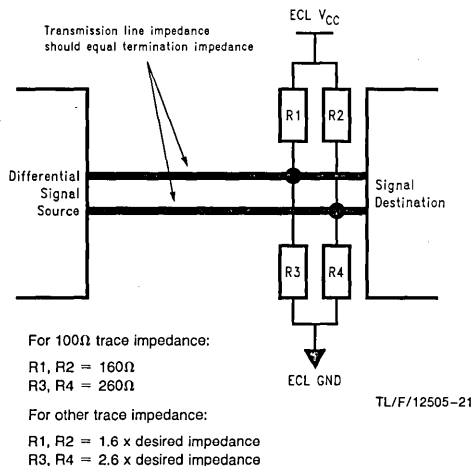


FIGURE 21. Thevenin Equivalent PECL Termination

### 8.4 Power Supply Isolation and Filtering

Proper power supply isolation and adequate filtering will help to reduce system noise sensitivities. The DP83223 and DP83840 devices possess unique analog and digital circuitry that require careful isolation and filtering techniques. Figure 26 illustrates the  $V_{CC}$  and Ground isolation connections recommended for each port within a typical 100BASE-TX repeater design. Recommended filtering for the DP83B50 is also provided.

Due to power handling limitations of the Ferrite Beads, it is recommended that each port be configured as illustrated in Figure 26.

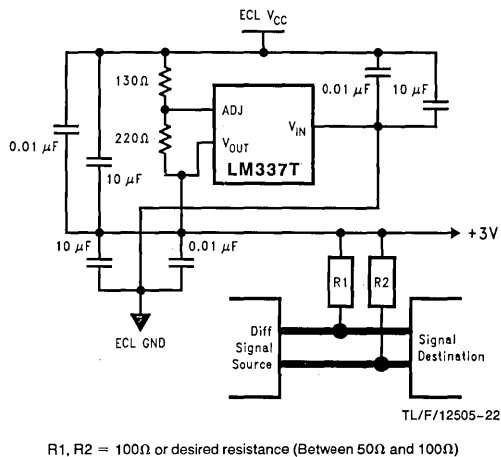


FIGURE 22. Direct PECL Termination



FIGURE 23. Board Layer Assignment Recommendation

TL/F/12505-23

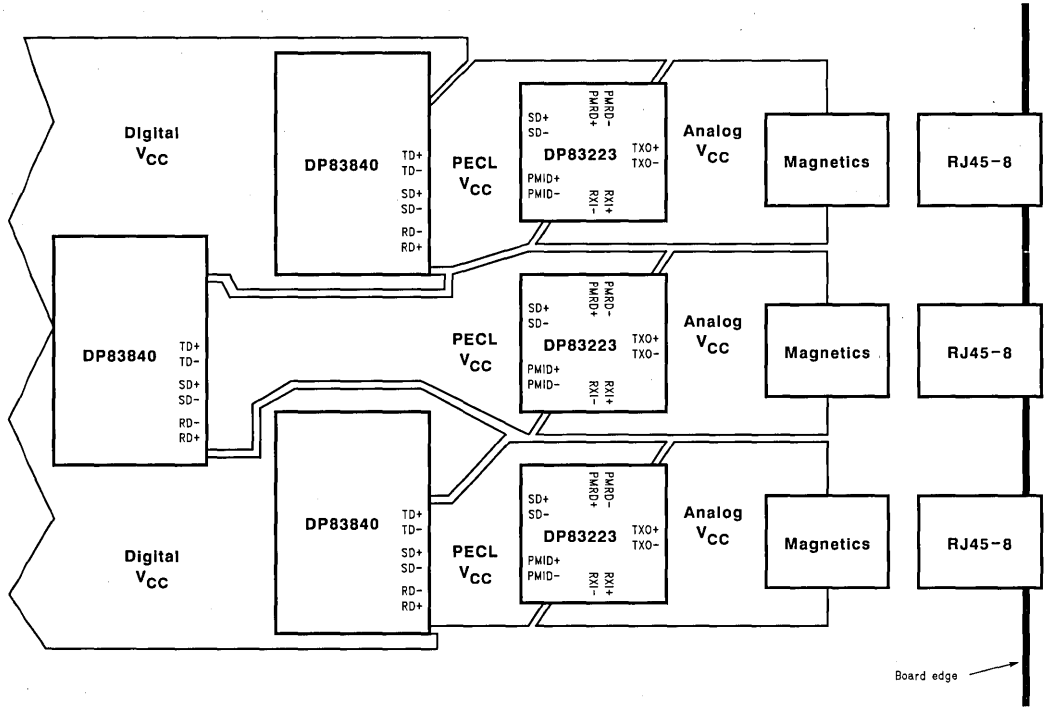


FIGURE 24. VCC Plane Isolation Example for 100BASE-TX Repeater

TL/F/12505-24

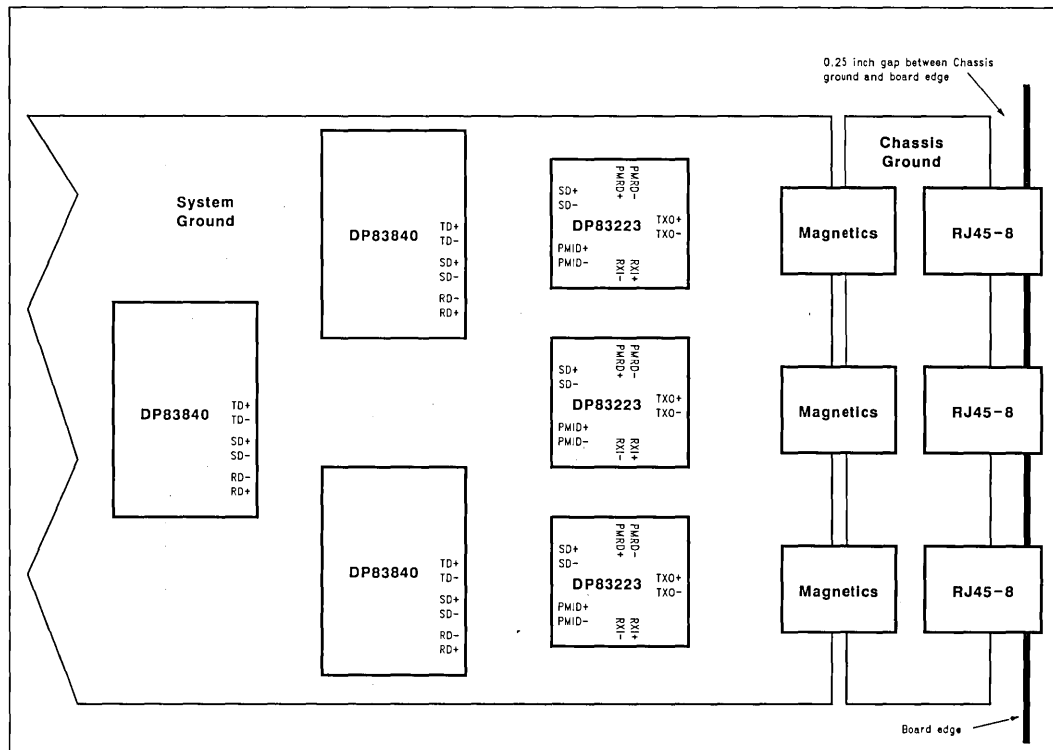


FIGURE 25. GND Plane Isolation Example for 100BASE-TX Repeater

TL/F/12505-25

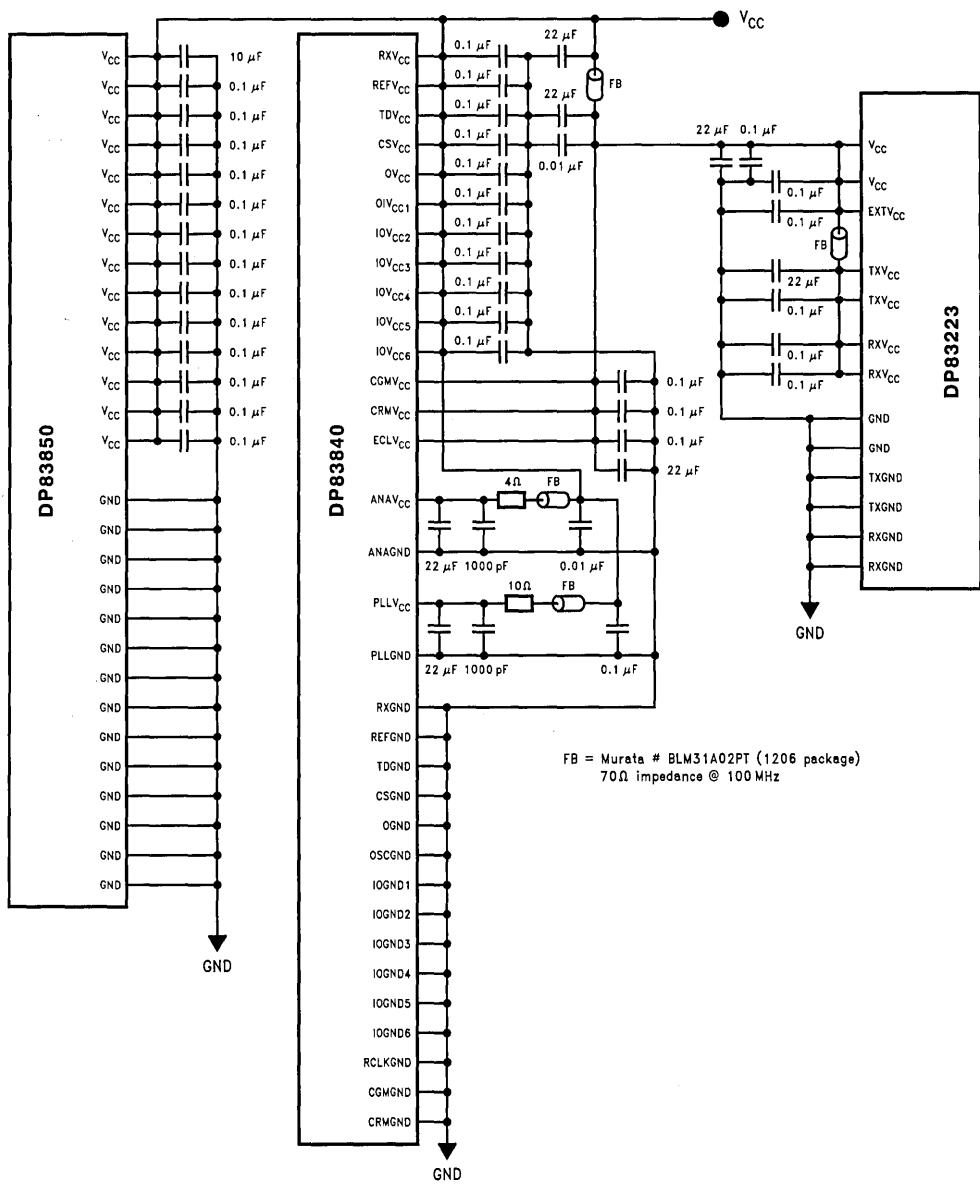


FIGURE 26. Power Supply Isolation and Filtering Recommendation

TL/F/12505-26





# Understanding Integrated Circuit Package Power Capabilities

National Semiconductor  
Application Note 336  
Charles Carinalli  
Josip Huljev

## INTRODUCTION

The short and long term reliability of National Semiconductor's interface circuits, like any integrated circuit, is very dependent on its environmental condition. Beyond the mechanical/environmental factors, nothing has a greater influence on this reliability than the electrical and thermal stress seen by the integrated circuit. Both of these stress issues are specifically addressed on every interface circuit data sheet, under the headings of Absolute Maximum Ratings and Recommended Operating Conditions.

However, through application calls, it has become clear that electrical stress conditions are generally more understood than the thermal stress conditions. Understanding the importance of electrical stress should never be reduced, but clearly, a higher focus and understanding must be placed on thermal stress. Thermal stress and its application to interface circuits from National Semiconductor is the subject of this application note.

## FACTORS AFFECTING DEVICE RELIABILITY

Figure 1 shows the well known "bathtub" curve plotting failure rate versus time. Similar to all system hardware (mechanical or electrical) the reliability of interface integrated circuits conform to this curve. The key issues associated with this curve are infant mortality, failure rate, and useful life.

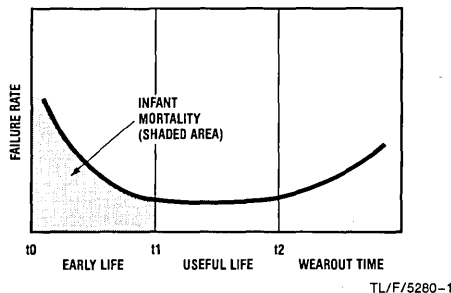


FIGURE 1. Failure Rate vs Time

Infant mortality, the high failure rate from time  $t_0$  to  $t_1$  (early life), is greatly influenced by system stress conditions other than temperature, and can vary widely from one application to another. The main stress factors that contribute to infant mortality are electrical transients and noise, mechanical maltreatment and excessive temperatures. Most of these failures are discovered in device test, burn-in, card assembly and handling, and initial system test and operation. Although important, much literature is available on the subject of infant mortality in integrated circuits and is beyond the scope of this application note.

Failure rate is the number of devices that will be expected to fail in a given period of time (such as, per million hours). The mean time between failure (MTBF) is the average time (in hours) that will be expected to elapse after a unit has failed before the next unit failure will occur. These two primary "units of measure" for device reliability are inversely related:

$$\text{MTBF} = \frac{1}{\text{Failure Rate}}$$

Although the "bathtub" curve plots the overall failure rate versus time, the useful failure rate can be defined as the percentage of devices that fail per-unit-time during the flat portion of the curve. This area, called the useful life, extends between  $t_1$  and  $t_2$  or from the end of infant mortality to the onset of wearout. The useful life may be as short as several years but usually extends for decades if adequate design margins are used in the development of a system.

Many factors influence useful life including: pressure, mechanical stress, thermal cycling, and electrical stress. However, die temperature during the device's useful life plays an equally important role in triggering the onset of wearout.

## FAILURE RATES vs TIME AND TEMPERATURE

The relationship between integrated circuit failure rates and time and temperature is a well established fact. The occurrence of these failures is a function which can be represented by the Arrhenius Model. Well validated and predominantly used for accelerated life testing of integrated circuits, the Arrhenius Model assumes the degradation of a performance parameter is linear with time and that MTBF is a function of temperature stress. The temperature dependence is an exponential function that defines the probability of occurrence. This results in a formula for expressing the lifetime or MTBF at a given temperature stress in relation to another MTBF at a different temperature. The ratio of these two MTBFs is called the acceleration factor  $F$  and is defined by the following equation:

$$F = \frac{X_1}{X_2} = \exp \left[ \frac{E}{K} \left( \frac{1}{T_2} - \frac{1}{T_1} \right) \right]$$

Where:  $X_1$  = Failure rate at junction temperature  $T_1$

$X_2$  = Failure rate at junction temperature  $T_2$

$T$  = Junction temperature in degrees Kelvin

$E$  = Thermal activation energy in electron volts (ev)

$K$  = Boltzman's constant

However, the dramatic acceleration effect of junction temperature (chip temperature) on failure rate is illustrated in a plot of the above equation for three different activation energies in *Figure 2*. This graph clearly demonstrates the importance of the relationship of junction temperature to device failure rate. For example, using the 0.99 eV line, a 30° rise in junction temperature, say from 130°C to 160°C, results in a 10 to 1 increase in failure rate.

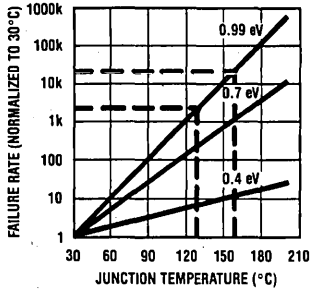


FIGURE 2. Failure Rate as a Function of Junction Temperature

TL/F/5280-2

**DEVICE THERMAL CAPABILITIES**

There are many factors which affect the thermal capability of an integrated circuit. To understand these we need to understand the predominant paths for heat to transfer out of the integrated circuit package. This is illustrated by *Figures 3* and *4*.

*Figure 3* shows a cross-sectional view of an assembled integrated circuit mounted into a printed circuit board.

*Figure 4* is a flow chart showing how the heat generated at the power source, the junctions of the integrated circuit

flows from the chip to the ultimate heat sink, the ambient environment. There are two predominant paths. The first is from the die to the die attach pad to the surrounding package material to the package lead frame to the printed circuit board and then to the ambient. The second path is from the package directly to the ambient air.

Improving the thermal characteristics of any stage in the flow chart of *Figure 4* will result in an improvement in device thermal characteristics. However, grouping all these characteristics into one equation determining the overall thermal capability of an integrated circuit/package/environmental condition is possible. The equation that expresses this relationship is:

$$T_J = T_A + P_D (\theta_{JA})$$

Where:  $T_J$  = Die junction temperature

$T_A$  = Ambient temperature in the vicinity device

$P_D$  = Total power dissipation (in watts)

$\theta_{JA}$  = Thermal resistance junction-to-ambient

$\theta_{JA}$ , the thermal resistance from device junction-to-ambient temperature, is measured and specified by the manufacturers of integrated circuits. National Semiconductor utilizes special vehicles and methods to measure and monitor this parameter. All interface circuit data sheets specify the thermal characteristics and capabilities of the packages available for a given device under specific conditions—these package power ratings directly relate to thermal resistance junction-to-ambient or  $\theta_{JA}$ .

Although National provides these thermal ratings, it is critical that the end user understand how to use these numbers to improve thermal characteristics in the development of his system using interface components.

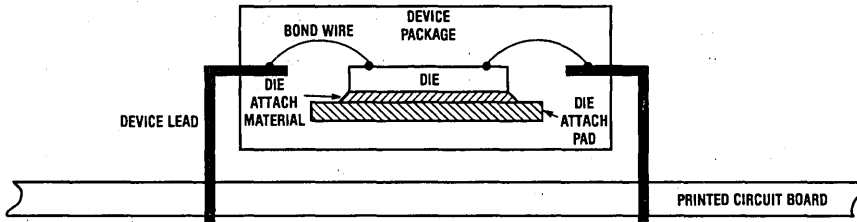


FIGURE 3. Integrated Circuit Soldered into a Printed Circuit Board (Cross-Sectional View)

TL/F/5280-3

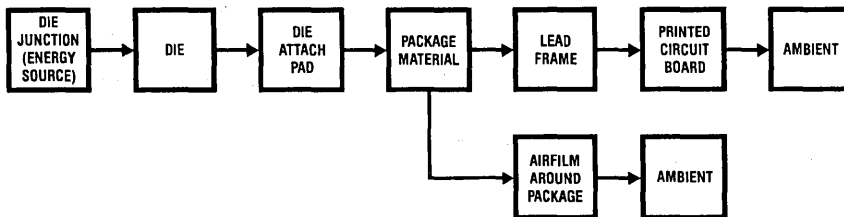


FIGURE 4. Thermal Flow (Predominant Paths)

TL/F/5280-4

**DETERMINING DEVICE OPERATING JUNCTION TEMPERATURE**

From the above equation the method of determining actual worst-case device operating junction temperature becomes straightforward. Given a package thermal characteristic,  $\theta_{JA}$ , worst-case ambient operating temperature,  $T_A(\max)$ , the only unknown parameter is device power dissipation,  $P_D$ . In calculating this parameter, the dissipation of the integrated circuit due to its own supply has to be considered, the dissipation within the package due to the external load must also be added. The power associated with the load in a dynamic (switching) situation must also be considered. For example, the power associated with an inductor or a capacitor in a static versus dynamic (say, 1 MHz) condition is significantly different.

The junction temperature of a device with a total package power of 600 mW at 70°C in a package with a thermal resistance of 63°C/W is 108°C.

$$T_J = 70^\circ\text{C} + (63^\circ\text{C}/\text{W}) \times (0.6\text{W}) = 108^\circ\text{C}$$

The next obvious question is, "how safe is 108°C?"

**MAXIMUM ALLOWABLE JUNCTION TEMPERATURES**

What is an acceptable maximum operating junction temperature is in itself somewhat of a difficult question to answer. Many companies have established their own standards based on corporate policy. However, the semiconductor industry has developed some defacto standards based on the device package type. These have been well accepted as numbers that relate to reasonable (acceptable) device lifetimes, thus failure rates.

National Semiconductor has adopted these industry-wide standards. For devices fabricated in a molded package, the maximum allowable junction temperature is 150°C. For these devices assembled in ceramic or cavity DIP packages, the maximum allowable junction temperature is 175°C. The numbers are different because of the differences in package types. The thermal strain associated with the die package interface in a cavity package is much less than that exhibited in a molded package where the integrated circuit chip is in direct contact with the package material.

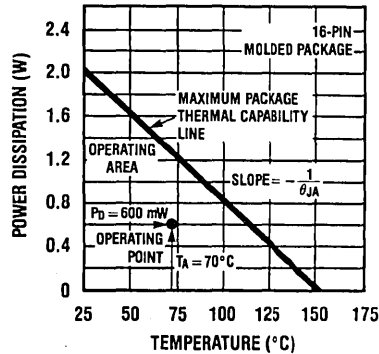
Let us use this new information and our thermal equation to construct a graph which displays the safe thermal (power) operating area for a given package type. Figure 5 is an example of such a graph. The end points of this graph are easily determined. For a 16-pin molded package, the maximum allowable temperature is 150°C; at this point no power dissipation is allowable. The power capability at 25°C is 1.98W as given by the following calculation:

$$P_D @ 25^\circ\text{C} = \frac{T_J(\max) - T_A}{\theta_{JA}} = \frac{150^\circ\text{C} - 25^\circ\text{C}}{63^\circ\text{C}/\text{W}} = 1.98\text{W}$$

The slope of the straight line between these two points is minus the inversion of the thermal resistance. This is referred to as the derating factor.

$$\text{Derating Factor} = -\frac{1}{\theta_{JA}}$$

As mentioned, Figure 5 is a plot of the safe thermal operating area for a device in a 16-pin molded DIP. As long as the intersection of a vertical line defining the maximum ambient temperature (70°C in our previous example) and maximum device package power (600 mW) remains below the maximum package thermal capability line the junction temperature will remain below 150°C—the limit for a molded package. If the intersection of ambient temperature and package power falls on this line, the maximum junction temperature will be 150°C. Any intersection that occurs above this line will result in a junction temperature in excess of 150°C and is not an appropriate operating condition.



TL/F/5260-5

**FIGURE 5. Package Power Capability vs Temperature**

The thermal capabilities of all interface circuits are expressed as a power capability at 25°C still air environment with a given derating factor. This simply states, for every degree of ambient temperature rise above 25°C, reduce the package power capability stated by the derating factor which is expressed in mW/°C. For our example—a  $\theta_{JA}$  of 63°C/W relates to a derating factor of 15.9 mW/°C.

**FACTORS INFLUENCING PACKAGE THERMAL RESISTANCE**

As discussed earlier, improving any portion of the two primary thermal flow paths will result in an improvement in overall thermal resistance junction-to-ambient. This section discusses those components of thermal resistance that can be influenced by the manufacturer of the integrated circuit. It also discusses those factors in the overall thermal resistance that can be impacted by the end user of the integrated circuit. Understanding these issues will go a long way in understanding chip power capabilities and what can be done to insure the best possible operating conditions and, thus, best overall reliability.

### Die Size

Figure 6 shows a graph of our 16-pin DIP thermal resistance as a function of integrated circuit die size. Clearly, as the chip size increases the thermal resistance decreases—this relates directly to having a larger area with which to dissipate a given power.

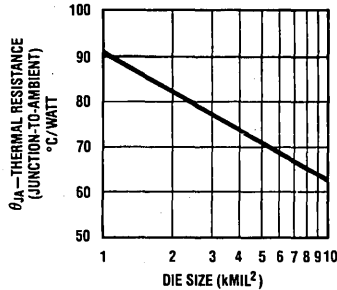


FIGURE 6. Thermal Resistance vs Die Size

TL/F/5280-6

### Lead Frame Material

Figure 7 shows the influence of lead frame material (both die attach and device pins) on thermal resistance. This graph compares our same 16-pin DIP with a copper lead frame, a Kovar lead frame, and finally an Alloy 43 type lead frame—these are lead frame materials commonly used in the industry. Obviously the thermal conductivity of the lead frame material has a significant impact in package power capability. Molded interface circuits from National Semiconductor use the copper lead frame exclusively.

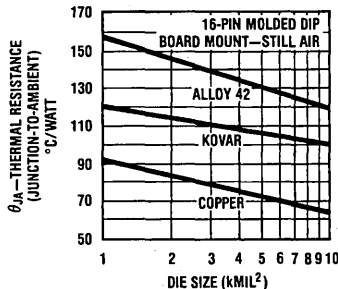


FIGURE 7. Thermal Resistance vs Lead Frame Material

TL/F/5280-7

### Board vs Socket Mount

One of the major paths of dissipating energy generated by the integrated circuit is through the device leads. As a result of this, the graph of Figure 8 comes as no surprise. This compares the thermal resistance of our 16-pin package soldered into a printed circuit board (board mount) compared to the same package placed in a socket (socket mount). Adding a socket in the path between the PC board and the device adds another stage in the thermal flow path, thus increasing the overall thermal resistance. The thermal capabilities of National Semiconductor's interface circuits are specified assuming board mount conditions. If the devices are placed in a socket the thermal capabilities should be reduced by approximately 5% to 10%.

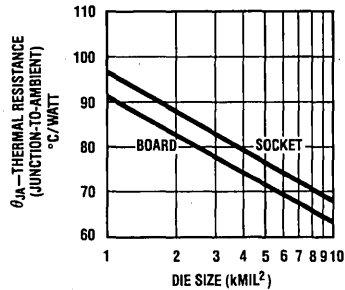


FIGURE 8. Thermal Resistance vs Board or Socket Mount

TL/F/5280-8

### Air Flow

When a high power situation exists and the ambient temperature cannot be reduced, the next best thing is to provide air flow in the vicinity of the package. The graph of Figure 9 illustrates the impact this has on thermal resistance. This graph plots the relative reduction in thermal resistance normalized to the still air condition for our 16-pin molded DIP. The thermal ratings on National Semiconductor's interface circuits data sheets relate to the still air environment.

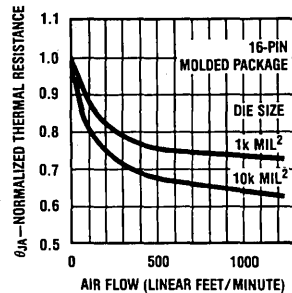


FIGURE 9. Thermal Resistance vs Air Flow

TL/F/5280-9

### Other Factors

A number of other factors influence thermal resistance. The most important of these is using thermal epoxy in mounting ICs to the PC board and heat sinks. Generally these techniques are required only in the very highest of power applications.

Some confusion exists between the difference in thermal resistance junction-to-ambient ( $\theta_{JA}$ ) and thermal resistance junction-to-case ( $\theta_{JC}$ ). The best measure of actual junction temperature is the junction-to-ambient number since nearly all systems operate in an open air environment. The only situation where thermal resistance junction-to-case is important is when the entire system is immersed in a thermal bath and the environmental temperature is indeed the case temperature. This is only used in extreme cases and is the exception to the rule and, for this reason, is not addressed in this application note.

## NATIONAL SEMICONDUCTOR PACKAGE CAPABILITIES

Figures 10 and 11 show composite plots of the thermal characteristics of the most common package types in the National Semiconductor Interface Circuits product family. Figure 10 is a composite of the copper lead frame molded package. Figure 11 is a composite of the ceramic (cavity) DIP using poly die attach. These graphs represent board mount still air thermal capabilities. Another, and final, thermal resistance trend will be noticed in these graphs. As the number of device pins increase in a DIP the thermal resistance decreases. Referring back to the thermal flow chart, this trend should, by now, be obvious.

### RATINGS ON INTERFACE CIRCUITS DATA SHEETS

In conclusion, all National Semiconductor Interface Products define power dissipation (thermal) capability. This information can be found in the Absolute Maximum Ratings section of the data sheet. The thermal information shown in this application note represents average data for characterization of the indicated package. Actual thermal resistance can vary from  $\pm 10\%$  to  $\pm 15\%$  due to fluctuations in assembly quality, die shape, die thickness, distribution of heat sources on the die, etc. The numbers quoted in the interface data sheets reflect a 15% safety margin from the average num-

bers found in this application note. Insuring that total package power remains under a specified level will guarantee that the maximum junction temperature will not exceed the package maximum.

The package power ratings are specified as a maximum power at 25°C ambient with an associated derating factor for ambient temperatures above 25°C. It is easy to determine the power capability at an elevated temperature. The power specified at 25°C should be reduced by the derating factor for every degree of ambient temperature above 25°C. For example, in a given product data sheet the following will be found:

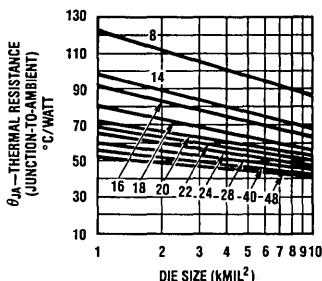
|                                    |         |
|------------------------------------|---------|
| Maximum Power Dissipation* at 25°C |         |
| Cavity Package                     | 1509 mW |
| Molded Package                     | 1476 mW |

\* Derate cavity package at 10 mW/°C above 25°C; derate molded package at 11.8 mW/°C above 25°C.

If the molded package is used at a maximum ambient temperature of 70°C, the package power capability is 945 mW.

$$P_D @ 70^\circ\text{C} = 1476 \text{ mW} - (11.8 \text{ mW}/^\circ\text{C}) \times (70^\circ\text{C} - 25^\circ\text{C}) \\ = 945 \text{ mW}$$

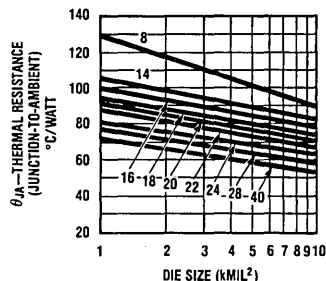
**Molded (N Package) DIP\*  
Copper Leadframe—HTP  
Die Attach Board Mount—  
Still Air**



\*Packages from 8- to 20-pin 0.3 mil width TL/F/5280-10  
22-pin 0.4 mil width  
24- to 40-pin 0.6 mil width

**FIGURE 10. Thermal Resistance vs Die Size vs Package Type (Molded Package)**

**Cavity (J Package) DIP\*  
Poly Die Attach Board  
Mount—Still Air**



\*Packages from 8- to 20-pin 0.3 mil width TL/F/5280-11  
22-pin 0.4 mil width  
24- to 48-pin 0.6 mil width

**FIGURE 11. Thermal Resistance vs Die Size vs Package Type (Cavity Package)**

# Operating ECL from a Single Positive Supply

National Semiconductor  
Application Note 780  
John Davis



## INTRODUCTION

ECL is normally specified for operation with a negative  $V_{EE}$  power source and a negative  $V_{TT}$  termination supply. This is the optimum operating configuration for ECL but not the only one. Operating ECL from a positive  $V_{CC}$  supply is a practical alternative that is gaining in popularity. Positive referenced ECL, or PECL as it is referred to, has been implemented in various mixed signal ASIC for use in the Video Graphics and Communications fields and is used in clock distribution as well. New single supply translator chips are becoming available to facilitate the interface of PECL logic levels to TTL and back again. Logic designers who strive for maximum speed in a system, now can easily replace sections of TTL logic with ECL and operate in PECL fashion from the common TTL  $V_{CC}$  supply.

## STANDARD NEGATIVE SUPPLY ECL OPERATION AND WHY

Figure 1 shows F100K logic elements operating in standard negative supply ECL configuration. The most positive potential is the primary voltage reference for ECL operation. Standard ECL input and output levels are therefore negative potentials referenced to the stable passive Ground (0V). The inherent F100K voltage compensation permits stable input and output levels over a broad range of  $V_{EE}$ 's; i.e.,  $-4.2$  to  $-5.7$  VDC for 300 Series F100K. Thus ECL logic operating from a  $-4.2$  V  $V_{EE}$  is compatible to logic operating from a  $-5.7$  V supply assuming both are referenced to a common 0V Ground.

Since ECL logic outputs only source currents that originate from the potential applied to its  $V_{CC}/V_{CCA}$  pins, the use of a 0V low impedance and low inductance ground potential is the optimum choice for operation. The use of a continuous

copper ground plane as the primary ECL reference is the ideal source for the high frequency transient currents demanded by the logic during switching. Note that despite the ideal nature of a ground plane as the primary ECL reference, when mixing TTL (or other noisy circuitry) into ECL systems, the recommendation is to reference the TTL to a separate ground plane. This is to keep the high transient TTL switching energy out of the primary ECL reference and preserve ECL noise margins.

When F100K ECL output signal interconnection lengths are direct and short enough, transmission line effects may be ignored and then only a RE output biasing resistor is required for logic operation. Please refer to section seven of the "F100K ECL Logic Databook and Design Guide" for a more detailed explanation of transmission line effects and ECL termination techniques. The RE resistor provides bias to keep the ECL emitter follower output transistor on for both high and low logic states. The RE resistor is normally connected between the ECL output and the most negative potential ( $V_{EE}$ ) thus permitting "single" supply operation.

The  $V_{EE}$  potential will ideally be distributed to the ECL logic from a power plane or bus which has low DC series resistance and low AC impedance. The low AC impedance is essential to supply the transient energy needed during switching. Although the inherent nature of ECL by design is to maintain essentially constant  $I_{EE}$  current even during switching, the charging and discharging of internal and external capacitances and the switching currents in the RE resistors place transient demands on  $V_{EE}$ . The degree to which the user can maintain complementary balance of ECL output loading will greatly influence the nature of the transient  $I_{EE}$  demands.

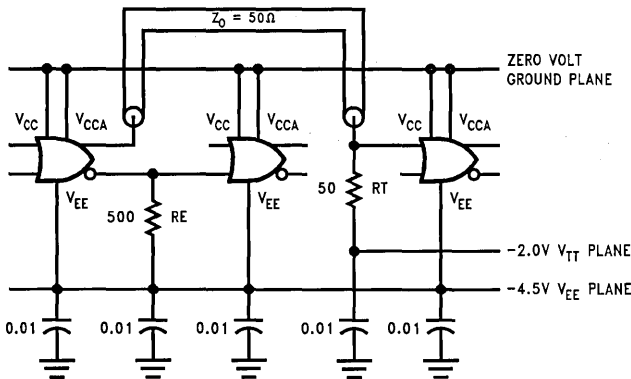


FIGURE 1. ECL Standard Operation from a Negative  $V_{EE}$  Supply

## F100K 300 Series Voltage Levels Specified for Standard Negative $V_{EE}$ Supply Operation

| Level    | Min    | Typ    | Max    | Units |
|----------|--------|--------|--------|-------|
| $V_{OH}$ | -1.025 | -0.955 | -0.87  | VDC   |
| $V_{IH}$ | -1.165 |        | -0.87  | VDC   |
| $V_{BB}$ |        | -1.320 |        | VDC   |
| $V_{IL}$ | -1.83  |        | -1.475 | VDC   |
| $V_{OL}$ | -1.83  | -1.705 | -1.62  | VDC   |

Conditions:  $V_{CC}/V_{CCA} = 0.0$  VDC Ground  
 $V_{EE} = -4.2$  to  $-5.7$  VDC  
 $R_T = 50\Omega$ ;  $V_{TT} = -2.0$  VDC  
 All Levels W.R.T. Ground

TL/F/10919-1

The usual recommendation for the  $V_{EE}$  plane is to bypass every ECL device at its  $V_{EE}$  pin with a good RF quality ceramic capacitor. The point at which the RE resistors return to the  $V_{EE}$  plane should also be bypassed particularly if it is a single return from a multiple resistor R-PAK. Values from  $0.01 \mu\text{F}$  to  $0.10 \mu\text{F}$  of the "High K Class II Dielectric" ceramic Z5U grade capacitor are recommended for commercial applications. The lower series inductance inherent in the leadless chip style capacitor is preferred over leaded types for highest frequency performance. The "Mid K Class II Dielectric" ceramic X7R grade capacitor offers acceptable bypass operating characteristics over the broader temperature range of  $-55^\circ\text{C}$  to  $+125^\circ\text{C}$ .

Bulk bypassing of the  $V_{EE}$  plane with a  $1 \mu\text{F}$  to  $10 \mu\text{F}$  is recommended at the point where the  $V_{EE}$  supply connects to the plane. Aluminum or Tantalum Electrolytic capacitors are usually used for bulk bypassing. Miniaturized surface mount Electrolytic capacitors are available for use in high density component applications.

In typical ECL system designs, some inter-connection lengths will exceed the critical values and force the consideration of transmission line effects. The most common high performance and power efficient termination scheme requires the use of a negative  $2.0\text{V}$   $V_{TT}$  termination supply. A single RT resistor in conjunction with the  $V_{TT}$  supply will terminate each output's transmission line in its characteristic impedance and will also provide optimum bias to the ECL output transistor.

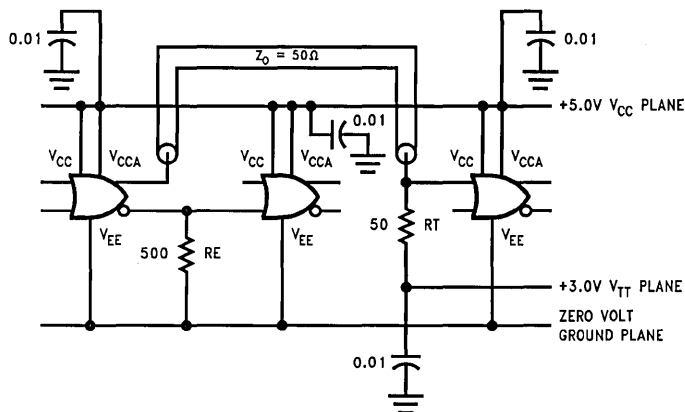
The  $V_{TT}$  potential will ideally be distributed to the RT terminators from a power plane which has low DC series resist-

ance and low AC impedance. The low AC impedance is essential to supply the transient energy in the termination resistors during switching. Bypassing  $V_{TT}$  wherever RT resistors return to the  $V_{TT}$  plane is essential to maintaining the low AC impedance of the plane. Capacitor recommendations for bypassing  $V_{TT}$  are the same as for  $V_{EE}$  above.

The regulation of the  $V_{TT}$  supply is not critical. A variation of  $\pm 5\%$  from nominal causes typically only  $\pm 12 \text{ mV}$  variation in output levels for  $50\Omega$  terminations or  $\pm 7 \text{ mV}$  variation for  $100\Omega$  terminations. Note that in standard ECL configuration, the  $V_{TT}$  supply need only sink current into its negative terminal (single ended  $V_{TT}$  operation with positive terminal grounded).  $V_{TT}$  here will typically be a simple series regulated supply. If the need for single negative supply operation is paramount, a less power efficient Thevenin termination scheme can be used between the  $V_{CC}/V_{CCA}$  and  $V_{EE}$  planes and selective use of series damping in conjunction with RE resistors may also be implemented.

### THE PECL TRANSFORMATION

Transforming ECL from negative supply to positive supply operations is conceptually quite easy. Just offset all standard ECL operating potentials by a positive amount equal to an absolute value within the normal  $V_{EE}$  operating range. For F100K 300 Series the normal  $V_{EE}$  range is  $-4.5$  to  $-5.7 \text{ VDC}$ . A  $5\text{V}$  offset fits nicely within the range and happens to match the nominal potential for TTL systems. Thus  $V_{EE}$  becomes the  $0\text{V}$  ground with  $V_{CC}/V_{CCA}$  offset to  $+5\text{V}$  and  $V_{TT}$  (if required) offset to  $+3 \text{ VDC}$ . Figure 2 illustrates the transformation (from Figure 1).



TL/F/10919-2

F100K 300 Series Voltage Levels Specified for Positive  $V_{CC}$  Supply [PECL] Operation

| Level    | Min              | Typ              | Max              | Units |
|----------|------------------|------------------|------------------|-------|
| $V_{OH}$ | $V_{CC} - 1.025$ | $V_{CC} - 0.955$ | $V_{CC} - 0.87$  | VDC   |
| $V_{IH}$ | $V_{CC} - 1.165$ |                  | $V_{CC} - 0.87$  | VDC   |
| $V_{BB}$ |                  | $V_{CC} - 1.320$ |                  | VDC   |
| $V_{IL}$ | $V_{CC} - 1.83$  |                  | $V_{CC} - 1.475$ | VDC   |
| $V_{OL}$ | $V_{CC} - 1.83$  | $V_{CC} - 1.705$ | $V_{CC} - 1.62$  | VDC   |

Conditions:  $V_{CC}/V_{CCA} = 4.2$  to  $5.7 \text{ VDC}$   
 $V_{EE} = 0.0\text{V}$  to Ground  
 $RT = 50\Omega$ ;  $V_{TT} = V_{CC} - 2.0 \text{ VDC}$   
 All Levels W.R.T. Ground

FIGURE 2. ECL Operation from a Positive  $V_{CC}$  Supply [PECL]

### CONSIDERATIONS FOR PECL OPERATION

All the considerations previously discussed for standard operation still apply; i.e., solid isolated and well bypassed reference planes, etc. Some additional considerations apply for PECL operation.

PECL input and output levels are referenced to the active positive  $V_{CC}$  rail that is variable and subject to line and load regulation. PECL level compatibility between sub-systems or systems can be difficult if precise  $V_{CC}$  distribution and accuracy are not maintained throughout. Differential PECL signal transmission and reception between systems may be necessary to ease the  $V_{CC}$  accuracy burden.

This active positive  $V_{CC}$  potential is the primary reference for PECL levels and the source of PECL switching currents. The distribution of  $V_{CC}$  to PECL logic is just as important as is the ground distribution to the standard ECL configuration.  $V_{CC}$  should be delivered from a continuous copper plane with liberal use of high frequency decoupling capacitors at each PECL device's  $V_{CC}/V_{CCA}$  pins.

If TTL or other noisy circuitry is to share the  $V_{CC}$ , a separate powerplane should be provided. TTL switching transients should be isolated from the PECL  $V_{CC}$  plane to preserve PECL noise immunity. Again, differential PECL operation may be warranted for situations where noise control is limited and good common mode noise rejection is required.

The various requirements for output termination and bias previously discussed for standard ECL applies directly to PECL operation. Note that the nominal  $+3V V_{TT}$  supply in PECL mode is required to sink current into its positive terminal (single ended  $V_{TT}$  operation with negative terminal grounded) from the emitter follower outputs through the RT resistors. A current sinking  $V_{TT}$  supply will be necessary if operated single ended to ground. The  $V_{TT}$  supply should track the  $V_{CC}$  supply keeping a nominal 2V offset to assure optimum biasing of the outputs.

The  $V_{EE}$  for PECL operation is 0V or ground potential and should be distributed from a continuous copper plane in consideration of handling the transients switching currents from the RE bias resistors. Although the PECL  $V_{EE}$  plane will be somewhat tolerant of TTL noise, the recommenda-

tion is to isolate TTL transient switching energy in a separate TTL ground plane.

### POWERPLANES

The dedication and organization of powerplanes is essential to successful ECL system design.

*Figure 3* illustrates an optimum powerplane implementation for Standard ECL operation on a printed circuit mother board in conjunction with TTL circuitry. *Figure 4* shows an optimum powerplane configuration for PECL operation. Note that the dedication and positioning of separate ECL and TTL powerplanes is intended to preserve ECL noise immunity when operating in a mixed signal environment.

| Copper Plane 1 | Signal                      |
|----------------|-----------------------------|
| 2              | TTL 0V Ground               |
| 3              | TTL +5V $V_{CC}$            |
| 4              | Auxiliary GND/Power/Thermal |
| 5              | ECL - 2V $V_{TT}$           |
| 6              | ECL - 4.5V $V_{EE}$         |
| 7              | ECL 0V Ground               |
| 8              | Signal                      |

**FIGURE 3. Powerplane Layout for Standard ECL Operation**

| Copper Plane 1 | Signal                      |
|----------------|-----------------------------|
| 2              | TTL 0V Ground               |
| 3              | TTL +5V $V_{CC}$            |
| 4              | Auxiliary GND/Power/Thermal |
| 5              | ECL + 3V $V_{TT}$           |
| 6              | ECL 0V $V_{EE}$ /Ground     |
| 7              | ECL + 5V $V_{CC}$           |
| 8              | Signal                      |

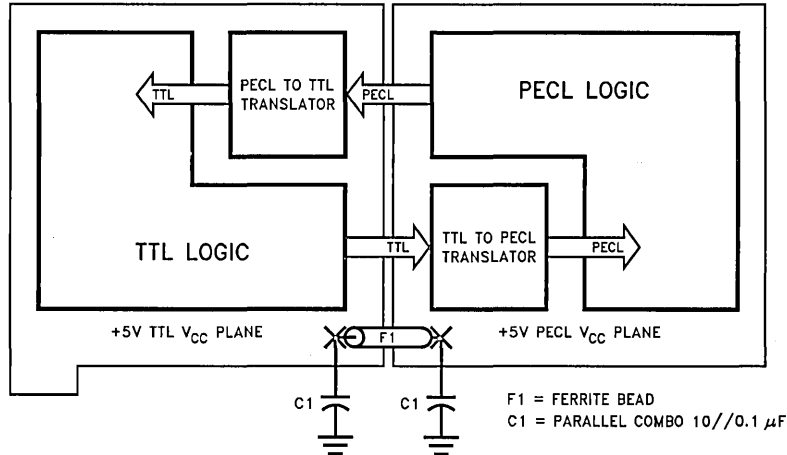
**FIGURE 4. Powerplane Layout for Positive Referenced ECL**



The optimum multiple powerplane approach may not be feasible for some designs. Logic and powerplane partitioning (islands) can be used to control noise when ECL and TTL must share the same powerplane. *Figure 5* illustrates the basic concept where areas of a system board are organized by logic type and share the same horizontal powerplane. Low pass filters are usually used to help isolate high frequency signals in sections of the shared plane.

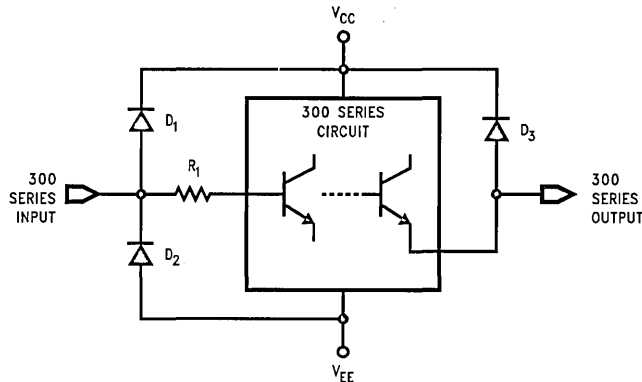
**POWER SUPPLY SEQUENCING CONSIDERATIONS**

In logic systems where multiple independent power supplies are used, or where two independently powered systems are connected logically, some consideration must be given to supply sequencing. This is particularly true for ECL/PECL logic due to placement of ESD (Electrostatic Discharge) protection diodes on the inputs and outputs. *Figure 6* shows the typical ESD diode placement in a F100K 300 Series device. *Figures 7a* and *7b* illustrate independently powered ECL driver and receiver operating with an independent ground referenced  $V_{TT}$  termination supply.



**FIGURE 5. Powerplane and Logic Partitioning**

TL/F/10919-5



**FIGURE 6. F100K 300 Series ESD Diode Circuit Placement**

TL/F/10919-6

When the devices (Figure 7a) are operated in Standard ECL fashion,  $V_{EE1}$  may be off while  $V_{EE2}$  and  $V_{TT}$  remain on without causing a forward bias potential on any of the ESD diodes. Note that both the true and complement outputs of the ECL1 driver will source logic one current simultaneously to the  $V_{TT}$  supply when  $V_{EE1}$  is off while  $V_{TT}$  remains on. Emitter follower transistors of ECL1 are biased on to a logic high level by the  $V_{TT}/RT$  even in absence of  $V_{EE1}$ . The potential for  $V_{TT}$  current overload exists under these circumstances.

When  $V_{EE2}$  is powered off and  $V_{TT}$  remains on, the low rail input ESD diode of ECL2 (connected to  $V_{EE2}$ ) will forward bias and conduct heavily as  $V_{TT}$  tries to re-power the  $V_{EE2}$  rail. The diode conduction will be limited by the RT resistor and the impedance of the off  $V_{EE2}$  supply in parallel with the ECL2 logic impedance. Although the ESD diode current density rating will typically support this current overstress, the recommendation is to avoid this by insuring that  $V_{EE2}$  and  $V_{TT}$  are ramped together and that  $V_{EE2}$  is never more positive than  $V_{TT}$  by 0.5V.

When the devices (Figure 7b) are operated in PECL fashion, there is a very clear forward bias hazard to ESD diodes when supplies are sequenced. If  $V_{CC2}$  is dropped before  $V_{CC1}$ , the positive referenced emitter followers of ECL1 will attempt to re-power up ECL2 through its high rail input ESD diode (connected to  $V_{CC}$ ). The ECL emitter follower outputs are low impedance voltage sources ( $6\Omega$  typical) and can source an incredible amount of current (greater than 200 mA each output). Thus  $V_{CC2}$  must never be more negative than  $V_{CC1}$  by 1.0V to avoid current overstress.

When  $V_{CC1}$  is powered off and  $V_{TT}$  and  $V_{EE2}$  remain on, the output ESD diode of ECL1 (connected to  $V_{CC1}$ ) will for-

ward bias and conduct heavily as  $V_{TT}$  tries to re-power the  $V_{CC1}$  rail. The diode conduction will be current limited by the RT resistor and the impedance of the off  $V_{CC1}$  supply in parallel with the ECL1 impedance. Although the ESD diode current density rating will typically support this current overstress, the recommendation is to avoid this by insuring that  $V_{TT}$  is never more positive than  $V_{CC1}$  by 0.5V.

If  $V_{CC1}$  and  $V_{CC2}$  are dropped while  $V_{TT}$  remains on, then  $V_{TT}$  tries to re-power both  $V_{CC}$  rails through the output ESD diode of ECL1 and the high rail input ESD diode of ECL2. The forward bias current is limited by the RT resistor and the  $V_{CC1}/V_{CC2}$  supply impedance in parallel with the collective logic impedance. This diode overstress is undesirable and should be avoided by insuring that  $V_{TT}$  is never more positive than  $V_{CC1}$  or  $V_{CC2}$  by more than 0.5V.

If  $V_{TT}$  is dropped before  $V_{CC1}$ , then increased load current can flow through the RT resistor from the emitter follower output of ECL1. Therefore  $V_{TT}$  ramping should be timed with  $V_{CC1}$  and  $V_{CC2}$ .

From the previous discussion, the most critical concern is that no PECL receiver should be powered down if driven directly by a powered up PECL driver without some form of current limiting. The inputs to the receiver must be current limited with external resistors of  $100\Omega$  or greater to be able to survive the overstress caused if  $V_{CC1}$  is ever permitted to be more positive than  $V_{CC2}$  by more than 1.0V. Although the use of current limiting resistors will alter the effective input edge rates and device propagation delays slightly, careful selection and placement of resistors will minimize device performance degradation. Use of surface mounted chip resistors located close to the input is recommended.

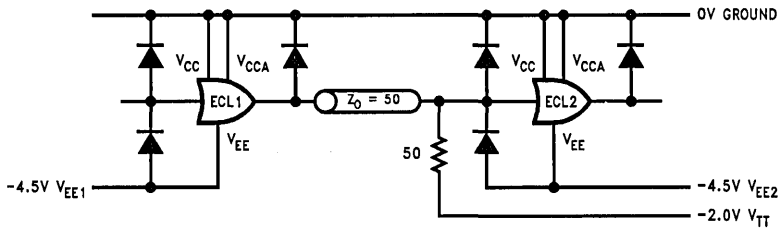


FIGURE 7a. ESD Diodes in Standard ECL Operation

TL/F/10919-3

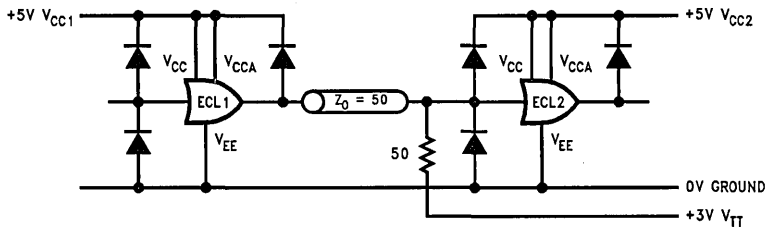


FIGURE 7b. ESD Diodes in PECL Operation

TL/F/10919-4

### DUAL SUPPLY TRANSLATORS—THE CONVENTIONAL APPROACH

Dual supply ECL-to-TTL and TTL-to-ECL IC translators have been in general use for several years. These devices perform the logic level translations between ECL operating from a negative  $V_{EE}$  supply and TTL operating from a positive  $V_{CC}$  supply. This approach naturally allows each logic

family to operate in their conventional and Data Book specified manner. System designers typically are most comfortable with the dual supply approach. This conventional method permits the use of the most familiar design practice for ECL and should easily yield reliable mixed signal system operation. The growing list of F100K 300 Series Dual Supply Translators, as shown in *Figure 8*, is testimony to the continued popularity and versatility of this approach.

| Features                                  | 100324 | 100325         | 100328 | 100329 | 100393 | 100395 | 100397 | 100398 |
|-------------------------------------------|--------|----------------|--------|--------|--------|--------|--------|--------|
| Data Bits                                 | 6      | 6              | 8      | 8      | 9      | 9      | 4      | 4      |
| ECL-to-TTL                                |        | X              | X      | X      | X      | X      | X      | X      |
| TTL-to-ECL                                | X      |                | X      | X      |        |        | X      | X      |
| Flow-Thru                                 | X      | X              |        |        |        |        |        |        |
| Latched                                   |        |                | X      |        | X      |        | X      | X      |
| Registered                                |        |                |        | X      |        | X      |        |        |
| ECL Differential Input                    |        | X <sup>1</sup> |        |        |        |        | X      | X      |
| ECL Differential Output                   | X      |                |        |        |        |        | X      | X      |
| ECL Output Drive ( $\Omega$ )             | 50     |                | 50     | 50     |        |        | 25     | 25     |
| ECL Cutoff (Hi Z)                         |        |                | X      | X      |        |        | X      | X      |
| TTL Output Drive (mA) ( $I_{OL}/I_{OH}$ ) |        | 20/-2          | 23/-3  | 24/-3  | 64/-15 | 64/-15 | 64/-15 | 64/-15 |
| TTL TRI-STATE®                            |        |                | X      | X      | X      | X      | X      | X      |
| ECL Control Pins                          |        |                | X      | X      | X      | X      | X      |        |
| TTL Control Pins                          | X      |                |        |        | X      |        |        | X      |
| TPD E to T (ns Max)                       |        | 4.8            | 5.9    | 7.7    | 5.3    | 6.4    | 5.8    | 5.8    |
| TPD T to E (ns Max)                       | 3.0    |                | 3.8    | 3.9    |        |        | 2.4    | 2.2    |
| $I_{EE}$ (mA Max)                         | -70    | -37            | -169   | -199   | -39    | -67    | -99    | -99    |
| $I_{EE}$ (mA Max) (Cutoff)                |        |                | -169   | -199   |        |        | -159   | -159   |
| $I_{CC}$ (mA Max)                         | 38     | 65             | 74     | 74     | 65     | 65     | 36     | 45     |

<sup>1</sup> $V_{BB}$  provided for Single-ended Operation

FIGURE 8. Table of F100K 300 Series Dual Supply Translators

### SINGLE SUPPLY TRANSLATORS—THE NEW WAVE APPROACH

Single Supply Translators that allow PECL-to-TTL or TTL-to-PECL interfaces are a recent addition to the F100K 300 Series ECL family. Development of these devices is motivated by the need for a convenient technique by which higher performance ECL logic can be integrated into existing TTL systems containing a single positive supply. These devices should also provide a vehicle for new lower cost designs of mixed signal single supply systems.

Figure 9 describes three such devices being offered in the F100K 300 Series family. The popularity of PECL operation

is expected to grow significantly as designers become more familiar with the technique. As interest and usage of Single Supply Translators increase, the family of this type of device can be expected to expand.

A simple illustration of the ease with which the Single Supply Translator can accomplish the interface from TTL to PECL and back to TTL is shown in Figure 10. Note that the translator devices have on chip  $V_{CC}$  partitions that facilitate the use of dual powerplanes for the preservation of ECL noise immunity. Differential operation on the PECL side of the translator is recommended to be used to maximize noise immunity. A  $V_{BB}$  reference voltage output is provided on the 100390 device to facilitate single ended operation.

| Features                                  | 100390         | 100391 | 100392 | 100389 |
|-------------------------------------------|----------------|--------|--------|--------|
| Data Bits                                 | 6              | 6      | 5      | 6      |
| ECL-to-TTL                                | X              |        |        |        |
| TTL-to-ECL                                |                | X      |        |        |
| CMOS-to-ECL                               |                |        | X      | X      |
| ECL Differential Input                    | X <sup>2</sup> |        |        |        |
| ECL Differential Output                   |                | X      | X      | X      |
| ECL Output Drive ( $\Omega$ )             |                | 50     | 25     | 50     |
| ECL Cutoff (Hi Z)                         |                |        | X      |        |
| TTL Output Drive (mA) ( $I_{OL}/I_{OH}$ ) | 24/-3          |        |        |        |
| TTL TRI-STATE                             | X              |        |        |        |
| TTL Control Pins                          | X              | X      |        |        |
| CMOS Control Pins                         |                |        | X      | X      |
| TPD E to T (ns Max)                       | 6.4            |        |        |        |
| TPD T to E (ns Max)                       |                | 1.7    |        |        |
| TPD C to E (ns Max)                       |                |        | TBD    | TBD    |
| $I_{EE}$ (mA Max) (Cutoff)                |                |        | TBD    |        |
| $I_{CC}$ (mA Max)                         | 48             | 60     | TBD    | TBD    |

<sup>2</sup> $V_{BB}$  provided for Single-ended Operation

FIGURE 9. Table of F100K 300 Series Single Supply Translators

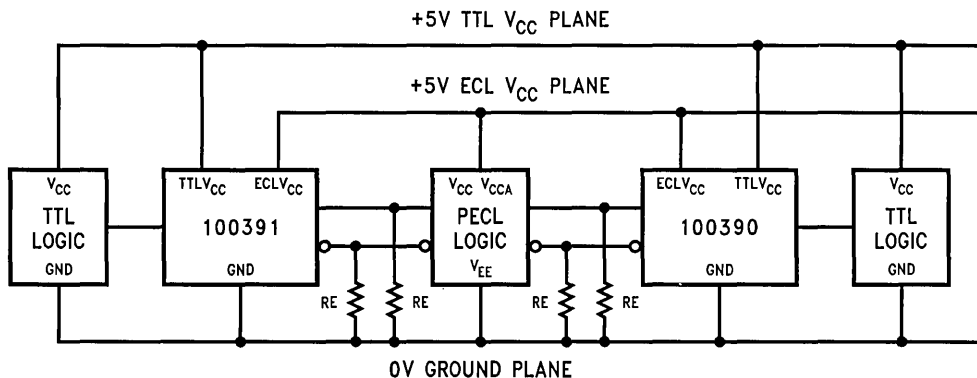


FIGURE 10. Use of Single Supply Translators

TL/F/10919-7

## Transmission Line Concepts

### Introduction

The interactions between wiring and circuitry in high-speed systems are more easily determined by treating the interconnections as transmission lines. A brief review of basic concepts is presented and simplified methods of analysis are used to examine situations commonly encountered in digital systems. Since the principles and methods apply to any type of logic circuit, normalized pulse amplitudes are used in sample waveforms and calculations.

### Simplifying Assumptions

For the great majority of interconnections in digital systems, the resistance of the conductors is much less than the input and output resistance of the circuits. Similarly, the insulating materials have very good dielectric properties. These circumstances allow such factors as attenuation, phase distortion, and bandwidth limitations to be ignored. With these simplifications, interconnections can be dealt with in terms of characteristic impedance and propagation delay.

### Characteristic Impedance

The two conductors that interconnect a pair of circuits have distributed series inductance and distributed capacitance between them, and thus constitute a transmission line. For any length in which these distributed parameters are constant, the pair of conductors have a characteristic impedance  $Z_0$ . Whereas quiescent conditions on the line are determined by the circuits and terminations,  $Z_0$  is the ratio of transient voltage to transient current passing by a point on the line when a signal charge or other electrical disturbance occurs. The relationship between transient voltage, transient current, characteristic impedance, and the distributed parameters is expressed as follows:

$$\frac{V}{I} = Z_0 = \sqrt{\frac{L_0}{C_0}} \quad (3-1)$$

where  $L_0$  = inductance per unit length,  $C_0$  = capacitance per unit length.  $Z_0$  is in ohms,  $L_0$  in Henries,  $C_0$  in Farads.

### Propagation Velocity

Propagation velocity  $v$  and its reciprocal, delay per unit length  $\delta$ , can also be expressed in terms of  $L_0$  and  $C_0$ . A consistent set of units is nanoseconds, microhenries and picofarads, with a common unit of length.

$$v = \frac{1}{\sqrt{L_0 C_0}} \quad \delta = \sqrt{L_0 C_0} \quad (3-2)$$

Equations 3-1 and 3-2 provide a convenient means of determining the  $L_0$  and  $C_0$ , of a line when delay, length and impedance are known. For a length  $l$  and delay  $T$ ,  $\delta$  is the ratio  $T/l$ . To determine  $L_0$  and  $C_0$ , combine Equations 3-1 and 3-2.

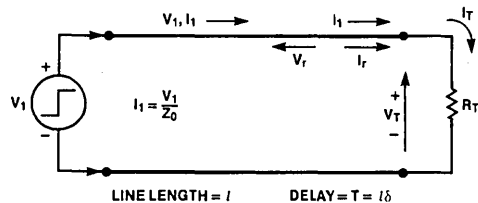
$$L_0 = \delta Z_0 \quad (3-3)$$

$$C_0 = \frac{\delta}{Z_0} \quad (3-4)$$

More formal treatments of transmission line characteristics, including loss effects, are available from many sources.<sup>1-3</sup>

### Termination and Reflection

A transmission line with a terminating resistor is shown in Figure 3-1. As indicated, a positive step function voltage travels from left to right. To keep track of reflection polarities, it is convenient to consider the lower conductor as the voltage reference and to think in terms of current flow in the top conductor only. The generator is assumed to have zero internal impedance. The initial current  $I_1$  is determined by  $V_1$  and  $Z_0$ .



TL/F/9900-1

**FIGURE 3-1. Assigned Polarities and Directions for Determining Reflections**

If the terminating resistor matches the line impedance, the ratio of voltage to current traveling along the line is matched by the ratio of voltage to current which must, by Ohm's law, always prevail at  $R_T$ . From the viewpoint of the voltage step generator, no adjustment of output current is ever required; the situation is as though the transmission line never existed and  $R_T$  had been connected directly across the terminals of the generator. From the  $R_T$  viewpoint, the only thing the line did was delay the arrival of the voltage step by the amount of time  $T$ .

When  $R_T$  is not equal to  $Z_0$ , the initial current starting down the line is still determined by  $V_1$  and  $Z_0$  but the final steady state current, after all reflections have died out, is determined by  $V_1$  and  $R_T$  (ohmic resistance of the line is assumed to be negligible). The ratio of voltage to current in the initial wave is not equal to the ratio of voltage to current demanded by  $R_T$ . Therefore, at the instant the initial wave arrives at  $R_T$ , another voltage and current wave must be generated so that Ohm's law is satisfied at the line-load interface. This *reflected* wave, indicated by  $V_r$  and  $I_r$  in Figure 3-1, starts to return toward the generator. Applying

### Termination and Reflection (Continued)

Kirchoff's laws to the end of the line at the instant the initial wave arrives, results in the following.

$$I_1 + I_r = I_T = \text{current into } R_T \quad (3-5)$$

Since only one voltage can exist at the end of the line at this instant of time, the following is true:

$$V_1 + V_r = V_T$$

thus 
$$I_r = \frac{V_T}{R_T} = \frac{V_1 + V_r}{R_T} \quad (3-6)$$

$$\text{also } I_1 = \frac{V_1}{Z_0} \text{ and } I_r = -\frac{V_r}{Z_0}$$

with the minus sign indicating that  $V_r$  is moving toward the generator.

Combining the foregoing relationships algebraically and solving for  $V_r$  yields a simplified expression in terms of  $V_1$ ,  $Z_0$  and  $R_T$ .

$$\frac{V_1}{Z_0} - \frac{V_r}{Z_0} = \frac{V_1 + V_r}{R_T} = \frac{V_1}{R_T} + \frac{V_r}{R_T}$$

$$V_1 \left( \frac{1}{Z_0} - \frac{1}{R_T} \right) = V_r \left( \frac{1}{R_T} + \frac{1}{Z_0} \right) \quad (3-7)$$

$$V_r = V_1 \left( \frac{R_T - Z_0}{R_T + Z_0} \right) = \rho_L V_1$$

The term in parenthesis is called the coefficient of reflection  $\rho$ . With  $R_T$  ranging between zero (shorted line) and infinity (open line), the coefficient ranges between  $-1$  and  $+1$  respectively. The subscript L indicates that  $\rho$  refers to the coefficient at the load end of the line.

Equation 3-7 expresses the amount of voltage sent back down the line, and since

$$V_T = V_1 + V_r \quad (3-8)$$

then 
$$V_T = V_1 (1 + \rho_L)$$

$V_T$  can also be determined from an expression which does not require the preliminary step of calculating  $\rho_L$ . Manipulating  $(1 + \rho_L)$  results in

$$1 + \rho_L = 1 + \frac{R_T - Z_0}{R_T + Z_0} = 2 \left( \frac{R_T}{R_T + Z_0} \right)$$

Substituting in Equation 3-8 gives

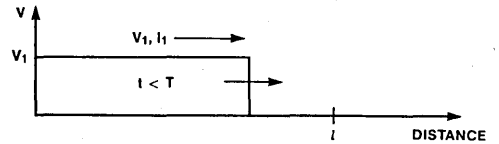
$$V_T = 2 \left( \frac{R_T}{R_T + Z_0} \right) V_1 \quad (3-9)$$

The foregoing has the same form as a simple voltage divider involving a generator  $V_1$  with internal impedance  $Z_0$  driving a load  $R_T$ , except that the amplitude of  $V_T$  is doubled.

The arrow indicating the direction of  $V_r$  in Figure 3-1 correctly indicates the  $V_r$  direction of travel, but the direction of  $I_r$  flow depends on the  $V_r$  polarity. If  $V_r$  is positive,  $I_r$  flows toward the generator, opposing  $I_1$ . This relationship between the polarity of  $V_r$  and the direction of  $I_r$  can be deduced by noting in Equation 3-7 that if  $V_r$  is positive it is because  $R_T$  is greater than  $Z_0$ . In turn, this means that the initial current  $I_r$  is larger than the final quiescent current, dictated by  $V_1$  and  $R_T$ . Hence,  $I_r$  must oppose  $I_1$  to reduce the line current to the final quiescent value. Similar reasoning shows that if  $V_r$  is negative,  $I_r$  flows in the same direction as  $I_1$ .

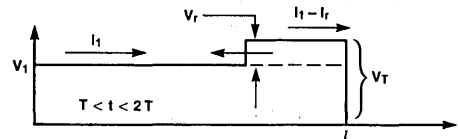
It is sometimes easier to determine the effect of  $V_r$  on line conditions by thinking of it as an independent voltage generator in series with  $R_T$ . With this concept, the direction of  $I_r$  is immediately apparent; its magnitude, however, is the ratio of  $V_r$  to  $Z_0$ , i.e.,  $R_T$  is already accounted for in the magnitude of  $V_r$ . The relationships between incident and reflected signals are represented in Figure 3-2 for both cases of mismatch between  $R_T$  and  $Z_0$ .

The incident wave is shown in Figure 3-2a, before it has reached the end of the line. In Figure 3-2b, a positive  $V_r$  is returning to the generator. To the left of  $V_r$ , the current is still  $I_1$ , flowing to the right, while to the right of  $V_r$  the net current in the line is the difference between  $I_1$  and  $I_r$ . In Figure 3-2c, the reflection coefficient is negative, producing a negative  $V_r$ . This, in turn, causes an increase in the amount of current flowing to the right behind the  $V_r$  wave.



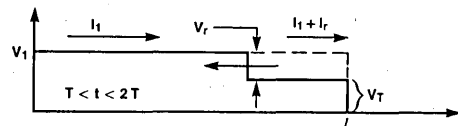
a. Incident Wave

TL/F/9900-2



b. Reflected Wave for  $R_T > Z_0$

TL/F/9900-3



c. Reflected Wave for  $R_T < Z_0$   
FIGURE 3-2. Reflections for  $R_T \neq Z_0$

TL/F/9900-4

## Source Impedance, Multiple Reflections

When a reflected voltage arrives back at the source (generator), the reflection coefficient at the source determines the response to  $V_r$ . The coefficient of reflection at the source is governed by  $Z_0$  and the source resistance  $R_S$ .

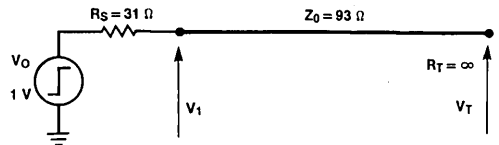
$$\rho_S = \frac{R_S - Z_0}{R_S + Z_0} \quad (3-10)$$

If the source impedance matches the line impedance, a reflected voltage arriving at the source is not reflected back toward the load end. Voltage and current on the line are stable with the following values.

$$V_T = V_1 + V_r \text{ and } I_T = I_1 - I_r \quad (3-11)$$

If neither source impedance nor terminating impedance matches  $Z_0$ , multiple reflections occur; the voltage at each end of the line comes closer to the final steady state value with each succeeding reflection. An example of a line mismatched on both ends is shown in *Figure 3-3*. The source is a step function of 1V amplitude occurring at time  $t_0$ . The initial value of  $V_1$  starting down the line is 0.75V due to the voltage divider action of  $Z_0$  and  $R_S$ . The time scale in the photograph shows that the line delay is approximately 6 ns. Since neither end of the line is terminated in its characteristic impedance, multiple reflections occur.

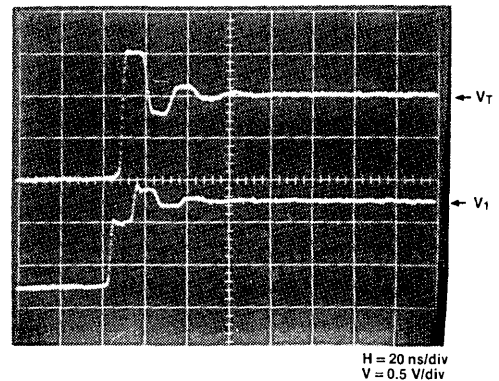
The amplitude and persistence of the ringing shown in *Figure 3-3* become greater with increasing mismatch between the line impedance and source and load impedances. Re-



TL/F/9900-5

$$\rho_S = \frac{31 - 93}{31 + 93} = -0.5 \quad \rho_L = \frac{\infty - 93}{\infty + 93} = +1$$

$$\text{Initially: } V_1 = \frac{Z_0}{Z_0 + R_S} \cdot V_0 = \frac{93}{124} \cdot 1 = 0.75V$$



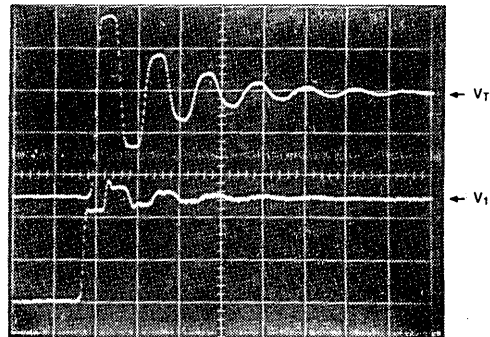
TL/F/9900-6

**FIGURE 3-3. Multiple Reflections Due to Mismatch at Load and Source**

ducing  $R_S$  (*Figure 3-3*) to 13Ω increases  $\rho_S$  to  $-0.75V$ , and the effects are illustrated in *Figure 3-4*. The initial value of  $V_T$  is 1.8V with a reflection of 0.9V from the open end. When this reflection reaches the source, a reflection of  $0.9V \times -0.75V$  starts back toward the open end. Thus, the second increment of voltage arriving at the open end is negative going. In turn, a negative-going reflection of  $0.9V \times -0.75V$  starts back toward the source. This negative increment is again multiplied by  $-0.75$  at the source and returned toward the open end. It can be deduced that the difference in amplitude between the first two positive peaks observed at the open end is

$$V_T - V'_T = (1 + \rho_L) V_1 - (1 + \rho_L) V_1 \rho_L^2 \rho_S^2 = (1 + \rho_L) V_1 (1 - \rho_L^2 \rho_S^2) \quad (3-12)$$

The factor  $(1 - \rho_L^2 \rho_S^2)$  is similar to the damping factor associated with lumped constant circuitry. It expresses the attenuation of successive positive or negative peaks of ringing.



H = 20 ns/div  
V = 0.4 V/div

TL/F/9900-7

**FIGURE 3-4. Extended Ringing when  $R_S$  of *Figure 3-3* is Reduced to 13Ω**

## Lattice Diagram

In the presence of multiple reflections, keeping track of the incremental waves on the line and the net voltage at the ends becomes a bookkeeping chore. A convenient and systematic method of indicating the conditions which combines magnitude, polarity and time utilizes a graphic construction called a lattice diagram.<sup>4</sup> A lattice diagram for the line conditions of *Figure 3-3* is shown in *Figure 3-5*.

The vertical lines symbolize discontinuity points, in this case the ends of the line. A time scale is marked off on each line in increments of  $2T$ , starting at  $t_0$  for  $V_1$  and  $T$  for  $V_T$ . The diagonal lines indicate the incremental voltages traveling between the ends of the line; solid lines are used for positive voltages and dashed lines for negative. It is helpful to write the reflection and transmission multipliers  $\rho$  and  $(1 + \rho)$  at each vertical line, and to tabulate the incremental and net voltages in columns alongside the vertical lines. Both the lattice diagram and the waveform photograph show that  $V_1$  and  $V_T$  asymptotically approach 1V, as they must with a 1V source driving an open-ended line.

## Lattice Diagram (Continued)

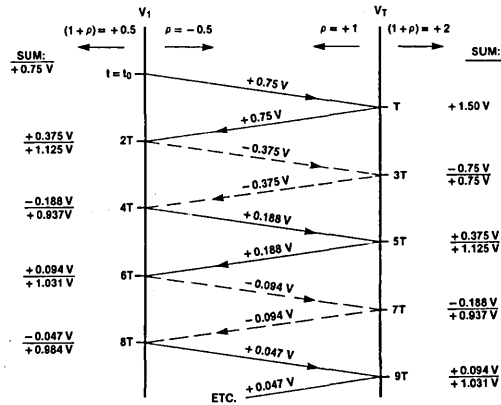


FIGURE 3-5. Lattice Diagram for the Circuit of Figure 3-3

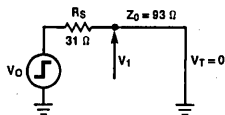
TL/F/9900-8

## Shorted Line

The open-ended line in Figure 3-3 has a reflection coefficient of +1 and the successive reflections tend toward the steady state conditions of zero line current and a line voltage equal to the source voltage. In contrast, a shorted line has a reflection coefficient of -1 and successive reflections must cause the line conditions to approach the steady state conditions of zero voltage and a line current determined by the source voltage and resistance.

Shorted line conditions are shown in Figure 3-6a with the reflection coefficient at the source end of the line also negative. A negative coefficient at both ends of the line means that any voltage approaching either end of the line is reflected in the opposite polarity. Figure 3-6b shows the response to an input step-function with a duration much longer than the line delay. The initial voltage starting down the line is about +0.75V, which is inverted at the shorted end and returned toward the source as -0.75V. Arriving back at the source end of the line, this voltage is multiplied by (1 + rho<sub>S</sub>), causing a -0.37V net change in V<sub>1</sub>. Concurrently, a reflected voltage of +0.37V (-0.75V times rho<sub>S</sub> of -0.5) starts back toward the shorted end of the line. The voltage at V<sub>1</sub> is reduced by 50% with each successive round trip of reflections, thus leading to the final condition of zero volts on the line.

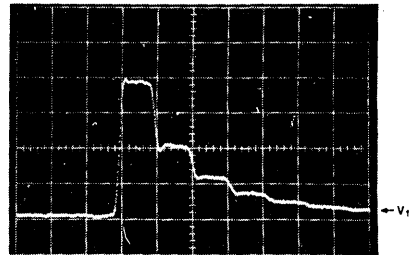
When the duration of the input pulse is less than the delay of the line, the reflections observed at the source end of the line constitute a train of negative pulses, as shown in Figure 3-6c. The amplitude decreases by 50% with each successive occurrence as it did in Figure 3-6b.



$\rho_S = -0.5$

TL/F/9900-9  
 $\rho_L = \frac{0 - 93}{0 + 93} = -1$

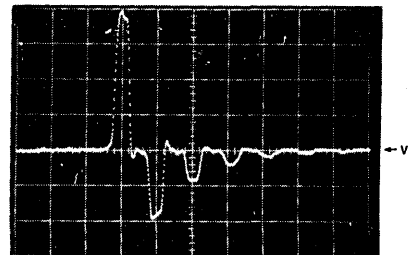
a. Reflection Coefficients for Shorted Line



H = 10 ns/div  
V = 0.2 V/div

TL/F/9900-10

b. Input Pulse Duration > Line Delay



H = 10 ns/div  
V = 0.2 V/div

TL/F/9900-11

c. Input Pulse Duration < Line Delay

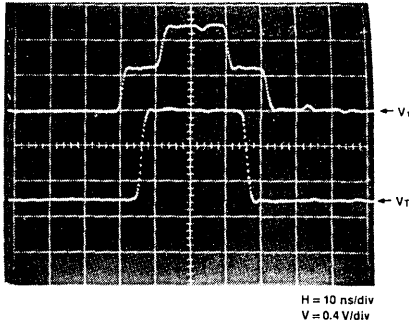
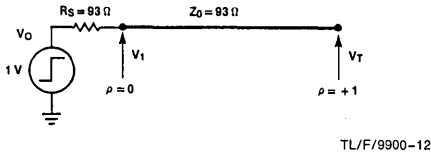
FIGURE 3-6. Reflections of Long and Short Pulses on a Shorted Line



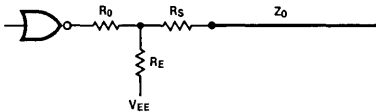
## Series Termination

Driving an open-ended line through a source resistance equal to the line impedance is called series termination. It is particularly useful when transmitting signals which originate on a PC board and travel through the backplane to another board, with the attendant discontinuities, since reflections coming back to the source are absorbed and ringing thereby controlled. *Figure 3-7* shows a  $93\Omega$  line driven from a  $1V$  generator through a source impedance of  $93\Omega$ . The photograph illustrates that the amplitude of the initial signal sent down the line is only half of the generator voltage, while the voltage at the open end of the line is doubled to full amplitude ( $1 + \rho_L = 2$ ). The reflected voltage arriving back at the source raises  $V_1$  to the full amplitude of the generator signal. Since the reflection coefficient at the source is zero, no further changes occur and the line voltage is equal to the generator voltage. Because the initial signal on the line is only half the normal signal swing, the loads must be connected at or near the end of the line to avoid receiving a 2-step input signal.

An ECL output driving a series terminated line requires a pull-down resistor to  $V_{EE}$ , as indicated in *Figure 3-8*. The resistor  $R_0$  shown in *Figure 3-8* symbolizes the output resistance of the ECL gate. The relationships between  $R_0$ ,  $R_S$ ,  $R_E$  and  $Z_0$  are discussed in Chapter 4.



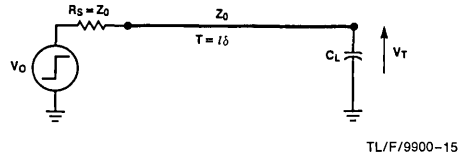
**FIGURE 3-7. Series Terminated Line and Waveforms**



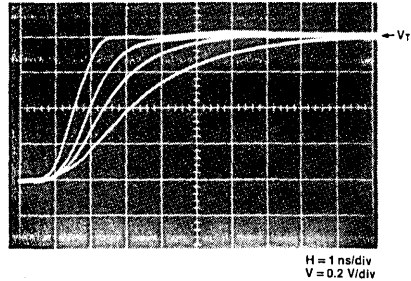
**FIGURE 3-8. ECL Element Driving a Series Terminated Line**

## Extra Delay with Termination Capacitance

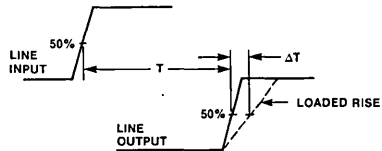
Designers should consider the effect of the load capacitance at the end of the line when using series termination. *Figure 3-9* shows how the output waveform changes with increasing load capacitance. *Figure 3-9b* shows the effect of load capacitances of 0, 12, 24, 48 pF. With no load, the delay between the 50% points of the input and output is just the line delay  $T$ . A capacitive load at the end of the line causes an extra delay  $\Delta T$  due to the increase in rise time of the output signal. The midpoint of the output is used as a criterion because the propagation delay of an ECL circuit is measured between the 50% points of the input and output signals.



**a. Series Terminated Line with Load Capacitance**



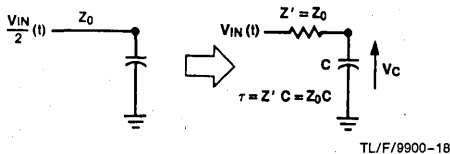
**b. Output Rise Time Increase with Increasing Load Capacitance**



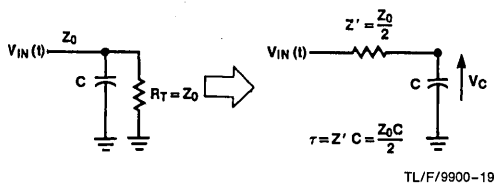
**c. Extra Delay  $\Delta T$  Due to Rise Time Increase**

**FIGURE 3-9. Extra Delay with Termination Capacitance**

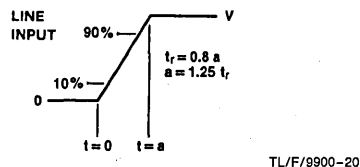
### Extra Delay with Termination Capacitance (Continued)



a. Thevenin Equivalent for Series Terminated Case



b. Thevenin Equivalent for Parallel Terminated Case



$$v_{in}(t) = \frac{V}{a} [tu(t) - (t-a)u(t-a)]$$

$$u(t) = \begin{cases} 0 & \text{for } t < 0 \\ 1 & \text{for } t > 0 \end{cases}$$

$$u(t-a) = \begin{cases} 0 & \text{for } t < a \\ 1 & \text{for } t > a \end{cases}$$

$$V_{IN}(S) = \frac{V}{as^2} (1 - e^{-as})$$

$$V_C(S) = \frac{V}{ar} \cdot \frac{1}{s^2(s + 1/\tau)} (1 - e^{-as})$$

$$v_C(t) = \frac{V}{a} [t - \tau(1 - e^{-t/\tau})] u(t) - \frac{V}{a} [(t-a) - \tau(1 - e^{-\frac{t-a}{\tau}})] u(t-a)$$

c. Equations for Input and Output Voltages

FIGURE 3-10. Determining the Effect of End-of-Line Capacitance

The increase in propagation delay can be calculated by using a ramp approximation for the incident voltage and characterizing the circuit as a fixed impedance in series with the load capacitance, as shown in *Figure 3-10*. One general solution serves both series and parallel termination cases by using an impedance  $Z'$  and a time constant  $\tau$ , defined in *Figure 3-10a* and *3-10b*. Calculated and observed increases in delay time to the 50% point show close agreement when  $\tau$  is less than half the ramp time. At large ratios of  $\tau/a$  (where  $a$  = ramp time), measured delays exceed calculated values by approximately 7%. *Figure 3-11*, based on measured values, shows the increase in delay to the 50% point as a function of the  $Z'C$  time constant, both normalized to the 10% to 90% rise time of the input signal. As an example of using the graph, consider a 100Ω series terminated line with 30 pF load capacitance at the end of the line and a no-load rise time of 3 ns for the input signal. From *Figure 3-10a*,  $Z'$  is equal to 100Ω; the ratio  $Z'C/t_r$  is 1. From the graph, the ratio  $\Delta T/t_r$  is 0.8. Thus the increase in the delay to the 50% point of the output waveform is 0.8  $t_r$ , or 2.4 ns, which is then added to the no-load line delay  $T$  to determine the total delay.

Had the 100Ω line in the foregoing example been parallel rather than series terminated at the end of the line,  $Z'$  would be 50Ω. The added delay would be only 1.35 ns with the same 30 pF loading at the end. The added delay would be only 0.75 ns if the line were 50Ω and parallel terminated. The various trade-offs involving type of termination, line impedance, and loading are important considerations for critical delay paths.

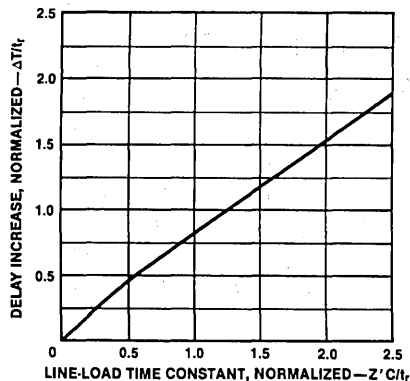


FIGURE 3-11. Increase in 50% Point Delay Due to Capacitive Loading at the End of the Line, Normalized to  $T_r$

TL/F/9900-21

## Distributed Loading Effects on Line Characteristics

When capacitive loads such as ECL inputs are connected along a transmission line, each one causes a reflection with a polarity opposite to that of the incident wave. Reflections from two adjacent loads tend to overlap if the time required for the incident wave to travel from one load to the next is equal to or less than the signal rise time.<sup>5</sup> *Figure 3-12a* illustrates an arrangement for observing the effects of capacitive loading, while *Figure 3-12b* shows an incident wave followed by reflections from two capacitive loads. The two capacitors causing the reflections are separated by a distance requiring a travel time of 1 ns. The two reflections return to the source 2 ns apart, since it takes 1 ns longer for the incident wave to reach the second capacitor and an additional 1 ns for the second reflection to travel back to the source. In the upper trace of *Figure 3-12b*, the input signal rise time is 1 ns and there are two distinct reflections, although the trailing edge of the first overlaps the leading edge of the second. The input rise time is longer in the middle trace, causing a greater overlap. In the lower trace, the 2 ns input rise time causes the two reflections to merge and appear as a single reflection which is relatively constant (at  $\approx -10\%$ ) for half its duration. This is about the same reflection that would occur if the 93 $\Omega$  line had a middle section with an impedance reduced to 75 $\Omega$ .

With a number of capacitors distributed all along the line of *Figure 3-12a*, the combined reflections modify the observed input waveform as shown in the top trace of *Figure 3-12c*. The reflections persist for a time equal to the 2-way line delay (15 ns), after which the line voltage attains its final value. The waveform suggests a line terminated with a resistance greater than its characteristic impedance ( $R_T >$

$Z_0$ ). This analogy is strengthened by observing the effect of reducing  $R_T$  from 93 $\Omega$  to 75 $\Omega$ , which leads to the middle waveform of *Figure 3-12c*. Note that the final (steady state) value of the line voltage is reduced by about the same amount as that caused by the capacitive reflections. In the lower trace of *Figure 3-12c* the source resistance  $R_S$  is reduced from 93 $\Omega$  to 75 $\Omega$ , restoring both the initial and final line voltage values to the same amplitude as the final value in the upper trace. From the standpoint of providing a desired signal voltage on the line and impedance matching at either end, the effect of distributed capacitive loading can be treated as a reduction in line impedance.

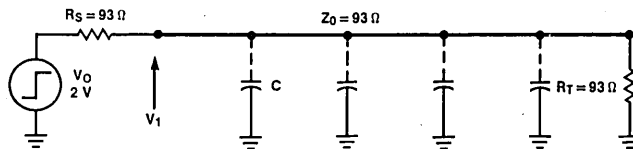
The reduced line impedance can be calculated by considering the load capacitance  $C_L$  as an increase in the intrinsic line capacitance  $C_0$  along that portion of the line where the loads are connected.<sup>6</sup> Denoting this length of line as  $l$ , the distributed value  $C_D$  of the load capacitance is as follows.

$$C_D = \frac{C_L}{l}$$

$C_D$  is then added to  $C_0$  in *Equation 3-1* to determine the reduced line impedance  $Z_0'$ .

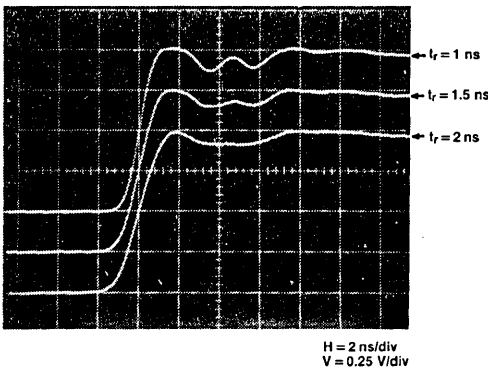
$$Z_0' = \sqrt{\frac{L_0}{C_0 + C_D}} = \sqrt{\frac{L_0}{C_0 \left(1 + \frac{C_D}{C_0}\right)}} \quad (3-13)$$

$$Z_0' + \frac{\sqrt{\frac{L_0}{C_0}}}{\sqrt{1 + \frac{C_D}{C_0}}} = \frac{Z_0}{\sqrt{1 + \frac{C_D}{C_0}}}$$



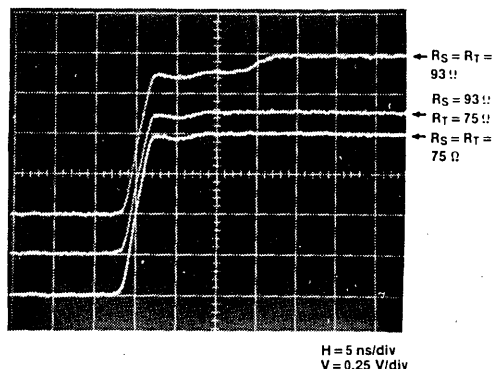
a. Arrangement for Observing Capacitive Loading Effects

TL/F/9900-22



b. Capacitive Reflections Merging as Rise Time Increases

TL/F/9900-23



c. Matching the Altered Impedance of a Capacitively Loaded Line

TL/F/9900-24

FIGURE 3-12. Capacitive Reflections and Effects on Line Characteristics

## Distributed Loading Effects on Line Characteristics (Continued)

In the example of *Figure 3-12c*, the total load capacitance is 33 pF while the total intrinsic line capacitance /  $C_0$  is 60 pF. (Note that the ratio  $C_D/C_0$  is the same as  $C_L/C_0$ .) The calculated value of the reduced impedance is thus

$$Z'_0 = \frac{93}{\sqrt{1 + \frac{33}{60}}} = \frac{93}{\sqrt{1.55}} = 75\Omega \quad (3-14)$$

This correlates with the results observed in *Figure 3-12c* when  $R_T$  and  $R_S$  are reduced to 75 $\Omega$ .

The distributed load capacitance also increases the line delay, which can be calculated from *Equation 3-2*.

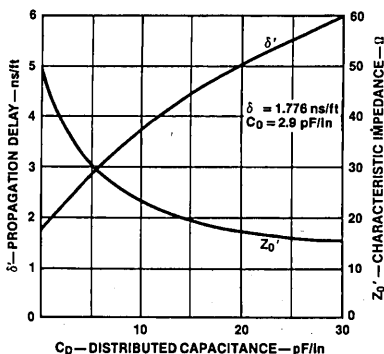
$$\begin{aligned} \delta' &= \sqrt{L_0(C_0 + C_D)} = \sqrt{L_0 C_0} \sqrt{1 + \frac{C_D}{C_0}} \\ &= \delta \sqrt{1 + \frac{C_D}{C_0}} \end{aligned} \quad (3-15)$$

The line used in the example of *Figure 3-12c* has an intrinsic delay of 6 ns and a loaded delay of 7.5 ns which checks with *Equation 3-15*.

$$1\delta' = 1\delta \sqrt{1.55} = 6\sqrt{1.55} = 7.5 \text{ ns} \quad (3-16)$$

*Equation 3-15* can be used to predict the delay for a given line and load. The ratio  $C_D/C_0$  (hence the loading effect) can be minimized for a given loading by using a line with a high intrinsic capacitance  $C_0$ .

A plot of  $Z'$  and  $\delta'$  for a 50 $\Omega$  line as a function of  $C_D$  is shown in *Figure 3-13*. This figure illustrates that relatively modest amounts of load capacitance will add appreciably to the propagation delay of a line. In addition, the characteristic impedance is reduced significantly.



TL/F/9900-25

**FIGURE 3-13. Capacitive Loading Effects on Line Delay and Impedance**

Worst case reflections from a capacitively loaded section of transmission line can be accurately predicted by using the modified impedance of *Equation 3-9*.<sup>6</sup> When a signal originates on an unloaded section of line, the effective reflection coefficient is as follows.

$$\rho = \frac{Z'_0 - Z_0}{Z'_0 + Z_0} \quad (3-17)$$

## Mismatched Lines

Reflections occur not only from mismatched load and source impedances but also from changes in line impedance. These changes could be caused by bends in coaxial cable, unshielded twisted-pair in contact with metal, or mismatch between PC board traces and backplane wiring. With the coax or twisted-pair, line impedance changes run about 5% to 10% and reflections are usually no problem since the percent reflection is roughly half the percent change in impedance. However, between PC board and backplane wiring, the mismatch can be 2 or 3 to 1. This is illustrated in *Figure 3-14* and analyzed in the lattice diagram of *Figure 3-15*. Line 1 is driven in the series terminated mode so that reflections coming back to the source are absorbed.

The reflection and transmission at the point where impedances differ are determined by treating the downstream line as though it were a terminating resistor. For the example of *Figure 3-14*, the reflection coefficient at the intersection of lines 1 and 2 for a signal traveling to the right is as follows.

$$\rho_{12} = \frac{Z_2 - Z_1}{Z_2 + Z_1} = \frac{93 - 50}{143} = +0.3 \quad (3-18)$$

Thus the signal reflected back toward the source and the signal continuing along line 2 are, respectively, as follows.

$$V_{1r} = \rho_{12} V_1 = +0.3V_1 \quad (3-19a)$$

$$V_2 = (1 + \rho_{12}) V_1 = +1.3V_1 \quad (3-19b)$$

At the intersection of lines 2 and 3, the reflection coefficient for signals traveling to the right is determined by treating  $Z_3$  as a terminating resistor.

$$\rho_{23} = \frac{Z_3 - Z_2}{Z_3 + Z_2} = \frac{39 - 93}{132} = -0.41 \quad (3-20)$$

When  $V_2$  arrives at this point, the reflected and transmitted signals are as follows.

$$\begin{aligned} V_{2r} &= \rho_{23} V_2 = -0.41 V_2 \\ &= (-0.41)(1.3) V_1 \\ &= -0.53 V_1 \end{aligned} \quad (3-21a)$$

$$\begin{aligned} V_3 &= (1 + \rho_{23}) V_2 = 0.59 V_2 \\ &= (0.59)(1.3) V_1 \\ &= 0.77 V_1 \end{aligned} \quad (3-21b)$$

Voltage  $V_3$  is doubled in magnitude when it arrives at the open-ended output, since  $\rho_L$  is +1. This effectively cancels the voltage divider action between  $R_S$  and  $Z_1$ .

$$\begin{aligned} V_4 &= (1 + \rho_L) V_3 = (1 + \rho_L)(1 + \rho_{23}) V_2 \\ &= (1 + \rho_L)(1 + \rho_{23})(1 + \rho_{12}) V_1 \\ &= (1 + \rho_L)(1 + \rho_{23})(1 + \rho_{12}) \frac{V_0}{2} \end{aligned} \quad (3-22)$$

$$V_4 = (1 + \rho_{23})(1 + \rho_{12}) V_0$$

Thus, *Equation 3-22* is the general expression for the initial step of output voltage for three lines when the input is series terminated and the output is open-ended.

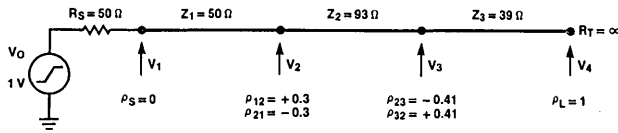
## Mismatched Lines (Continued)

Note that the reflection coefficients at the intersections of lines 1 and 2 and lines 2 and 3 in *Figure 3-15* have reversed signs for signals traveling to the left. Thus the voltage reflected from the open output and the signal reflecting back and forth on line 2 both contribute additional increments of output voltage in the same polarity as  $V_0$ . Lines 2 and 3 have the same delay time; therefore, the two aforementioned increments arrive at the output simultaneously at time  $5T$  on the lattice diagram (*Figure 3-15*).

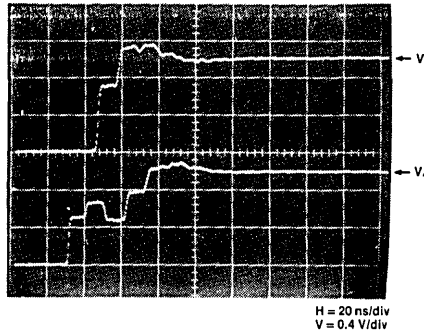
In the general case of series lines with different delay times, the vertical lines on the lattice diagram should be spaced apart in the ratio of the respective delays. *Figure 3-16* shows this for a hypothetical case with delay ratios 1:2:3. For a sequence of transmission lines with the highest im-

pedance line in the middle, at least three output voltage increments with the same polarity as  $V_0$  occur before one can occur of opposite polarity. On the other hand, if the middle line has the lowest impedance, the polarity of the second increment of output voltage is the opposite of  $V_0$ . The third increment of output voltage has the opposite polarity, for the time delay ratios of *Figure 3-16*.

When transmitting logic signals, it is important that the initial step of line output voltage pass through the threshold region of the receiving circuit, and that the next two increments of output voltage augment the initial step. Thus in a series terminated sequence of three mismatched lines, the middle line should have the highest impedance.

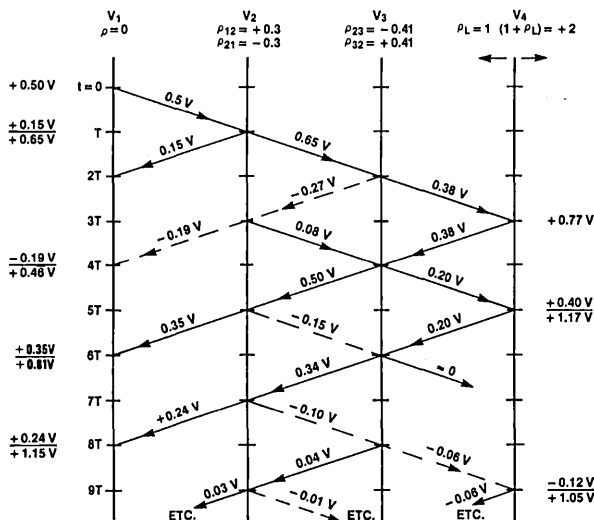


TL/F/9900-26



TL/F/9900-27

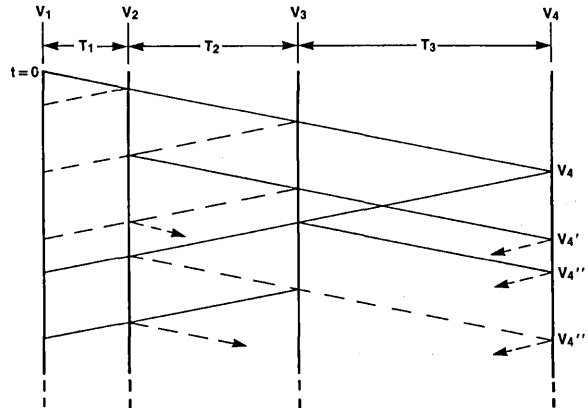
FIGURE 3-14. Reflections from Mismatched Lines



TL/F/9900-28

FIGURE 3-15. Lattice Diagram for the Circuit of *Figure 3-14*

## Mismatched Lines (Continued)



TL/F/9900-29

FIGURE 3-16. Lattice Diagram for Three Lines with Delay Ratios 1:2:3

## Rise Time versus Line Delay

When the 2-way line delay is less than the rise time of the input wave, any reflections generated at the end of the line are returned to the source before the input transition is completed. Assuming that the generator has a finite source resistance, the reflected wave adds algebraically to the input wave while it is still in transition, thereby changing the shape of the input. This effect is illustrated in *Figure 3-17*, which shows input and output voltages for several comparative values of rise time and line delay.

In *Figure 3-17b* where the rise time is much shorter than the line delay,  $V_1$  rises to an initial value of 1V. At time  $T$  later,  $V_T$  rises to 0.5V, i.e.,  $1 + \rho_L = 0.5$ . The negative reflection arrives back at the source at time  $2T$ , causing a net change of  $-0.4V$ , i.e.,  $(1 + \rho_S)(-0.5) = -0.4$ .

The negative coefficient at the source changes the polarity of the other 0.1V of the reflection and returns it to the end of the line, causing  $V_T$  to go positive by another 50 mV at time  $3T$ . The remaining 50 mV is inverted and reflected back to the source, where its effect is barely distinguishable as a small negative change at time  $4T$ .

In *Figure 3-17c*, the input rise time (0% to 100%) is increased to such an extent that the input ramp ends just as the negative reflection arrives back at the source end. Thus the input rise time is equal to  $2T$ .

The input rise time is increased to  $4T$  in *Figure 3-17d*, with the negative reflection causing a noticeable change in input slope at about its midpoint. This change in slope is more visible in the double exposure photo of *Figure 3-17e*, which shows  $V_1$  ( $t_r$  still set for  $4T$ ) with and without the negative reflection. The reflection was eliminated by terminating the line in its characteristic impedance.

The net input voltage at any particular time is determined by adding the reflection to the otherwise unaffected input. It must be remembered that the reflection arriving back at the input at a given time is proportional to the input voltage at a time  $2T$  earlier. The value of  $V_1$  in *Figure 3-17d* can be calculated by starting with the 1V input ramp.

$$V_1 = \frac{1}{t_r} \cdot t \quad \text{for } 0 \leq t \leq 4T \quad (3-23)$$

$$= 1V \quad \text{for } t \geq 4T$$

The reflection from the end of the line is

$$V_r = \frac{\rho_L(t - 2T)}{t_r} \quad (3-24)$$

the portion of the reflection that appears at the input is

$$V'_r = \frac{(1 + \rho_S)\rho_L(t - 2T)}{t_r} \quad (3-25)$$

the net value of the input voltage is the sum.

$$V'_1 = \frac{t}{t_r} + \frac{(1 + \rho_S)\rho_L(t - 2T)}{t_r} \quad (3-26)$$

The peak value of the input voltage in *Figure 3-17d* is determined by substituting values and letting  $t$  equal  $4T$ .

$$V'_1 = 1 + \frac{(0.8)(-0.5)(4T - 2T)}{t_r} \quad (3-27)$$

$$= 1 - 0.4(0.5) = 0.8V$$

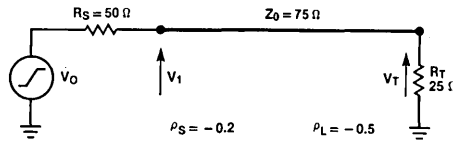
After this peak point, the input ramp is no longer increasing but the reflection is still arriving. Hence the net value of the input voltage decreases. In this example, the later reflections are too small to be detected and the input voltage is thus stable after time  $6T$ . For the general case of repeated reflections, the net voltage  $V_1(t)$  seen at the driven end of the line can be expressed as follows, where the signal caused by the generator is  $V_1(t)$ .

### Rise Time versus Line Delay (Continued)

$$\begin{aligned}
 V'_1(t) &= V_1(t) && \text{for } 0 < t < 2T \\
 V'_1(t) &= V_1(t) + (1 + \rho_S) \rho_L V_1(t-2T) && \text{for } 2T < t < 4T \\
 V'_1(t) &= V_1(t) + (1 + \rho_S) \rho_L V_1(t-2T) && + (1 + \rho_S) \rho_S \rho_L^2 V_1(t-4T) && (3-28) \\
 &&& \text{for } 4T < t < 6T \\
 V'_1(t) &= V_1(t) + (1 + \rho_S) \rho_L V_1(t-2T) && + (1 + \rho_S) \rho_S \rho_L^2 V_1(t-4T) && + (1 + \rho_S) \rho_S^2 \rho_L^3 V_1(t-6T) \\
 &&& \text{for } 6T < t < 8T, \text{ etc.}
 \end{aligned}$$

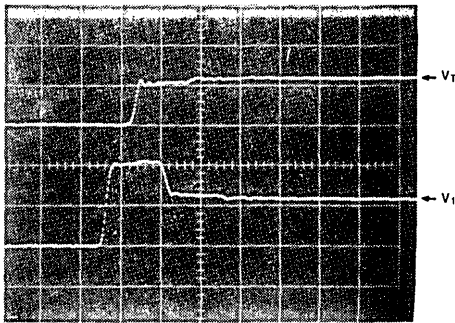
The voltage at the output end of the line is expressed in a similar manner.

$$\begin{aligned}
 V_T(t) &= 0 && \text{for } 0 < t < T \\
 V_T(t) &= (1 + \rho_L) V_1(t-T) && \text{for } T < t < 3T \\
 V_T(t) &= (1 + \rho_L) V_1(t-T) && + (1 + \rho_L) \rho_S \rho_L V_1(t-3T) && (3-29) \\
 &&& \text{for } 3T < t < 5T \\
 V_T(t) &= (1 + \rho_L) V_1(t-T) && + (1 + \rho_L) \rho_S \rho_L V_1(t-3T) && + (1 + \rho_L) \rho_S^2 \rho_L^2 V_1(t-5T) \\
 &&& \text{for } 5T < t < 7T, \text{ etc.}
 \end{aligned}$$



TL/F/9900-30

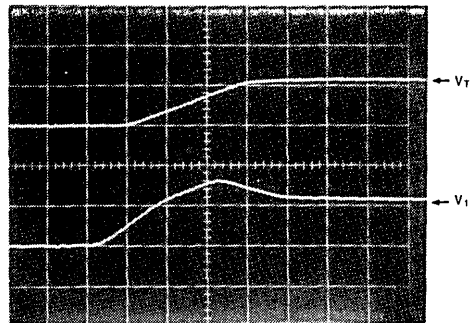
a. Test Arrangement for Rise Time Analysis



H = 10 ns/div  
V = 0.5 V/div

TL/F/9900-31

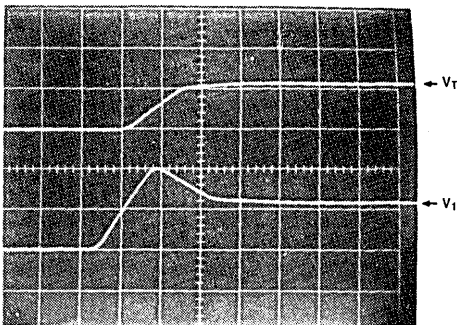
b. Line Voltages for  $t_r \ll T$



H = 10 ns/div  
V = 0.5 V/div

TL/F/9900-33

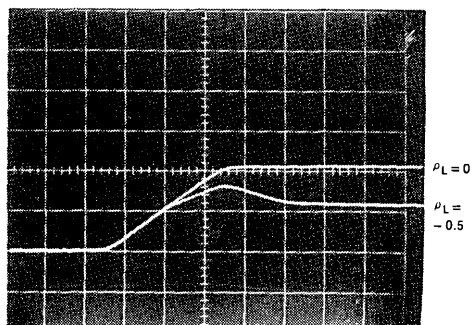
d. Line Voltages for  $t_r = 4T$



H = 10 ns/div  
V = 0.5 V/div

TL/F/9900-32

c. Line Voltages for  $t_r = 2T$



H = 10 ns/div  
V = 0.5 V/div

TL/F/9900-34

e. Input Voltage with and without Reflection

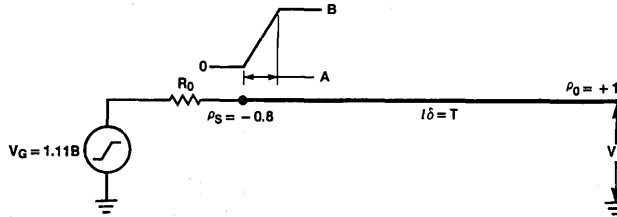
FIGURE 3-17. Line Voltages for Various Ratios of Rise Time to Line Delay

## Ringing

Multiple reflections occur on a transmission line when neither the signal source impedance nor the termination (load) impedance matches the line impedance. When the source reflection coefficient  $\rho_S$  and the load reflection coefficient  $\rho_L$  are of opposite polarity, the reflections alternate in polarity. This causes the signal voltage to oscillate about the final steady state value, commonly recognized as ringing.

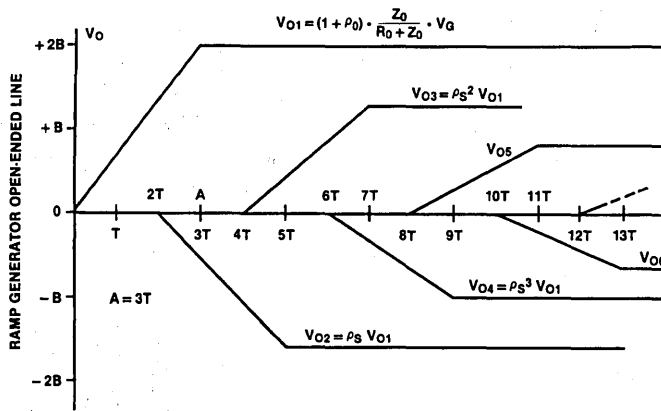
When the signal rise time is long compared to the line delay, the signal shape is distorted because the individual reflections overlap in time. The basic relationships among rise time, line delay, overshoot and undershoot are shown in a simplified diagram, *Figure 3-18*. The incident wave is a ramp of amplitude  $B$  and rise duration  $A$ . The reflection coefficient at the open-ended line output is  $+1$  and the source reflection coefficient is assumed to be  $-0.8$ , i.e.,  $R_0 = Z_0/9$ .

*Figure 3-18b* shows the individual reflections treated separately. Rise time  $A$  is assumed to be three times the line delay  $T$ . The time scale reference is the line output and the first increment of output voltage  $V_O$  rises to  $2B$  in the time interval  $A$ . Simultaneously, a positive reflection (not shown) of amplitude  $B$  is generated and travels to the source, whereupon it is multiplied by  $-0.8$  and returns toward the end of the line. This negative-going ramp starts at time  $2T$  (twice the line delay) and doubles to  $-1.6B$  at time  $2T + A$ . The negative-going increment also generates a reflection of amplitude  $-0.8B$  which makes the round trip to the source and back, appearing at time  $4T$  as a positive ramp rising to  $+1.28B$  at time  $4T + A$ . The process of reflection and re-reflection continues, and each successive increment changes in polarity and has an amplitude of 80% of the preceding increment.



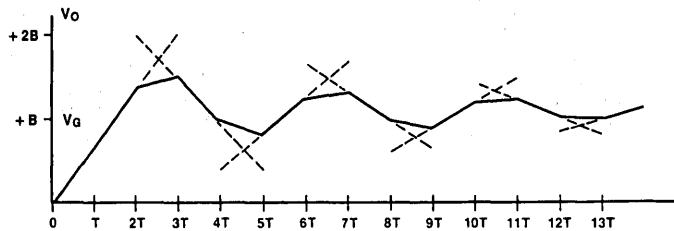
a. Ramp Generator Driving Open-Ended Line

TL/F/9900-35



b. Increments of Output Voltage Treated Individually

TL/F/9900-36



c. Net Output Signal Determined by Superposition

TL/F/9900-37

FIGURE 3-18. Basic Relationships Involved in Ringing



## Ringings (Continued)

In *Figure 3-18c*, the output increments are added algebraically by superposition. The starting point of each increment is shifted upward to a voltage value equal to the algebraic sum of the quiescent levels of all the preceding increments (i.e., 0, 2B, 0.4B, 1.6B, etc.). For time intervals when two ramps occur simultaneously, the two linear functions add to produce a third ramp that prevails during the overlap time of the two increments.

It is apparent from the geometric relationships, that if the ramp time A is less than twice the line delay, the first output increment has time to rise to the full 2B amplitude and the second increment reduces the net output voltage to 0.4B. Conversely, if the line delay is very short compared to the ramp time, the excursions about the final value  $V_G$  are small.

*Figure 3-18c* shows that the peak of each excursion is reached when the earlier of the two constituent ramps reaches its maximum value, with the result that the first peak occurs at time A. This is because the earlier ramp has a greater slope (absolute value) than the one that follows.

Actual waveforms such as produced by ECL or TTL do not have a constant slope and do not start and stop as abruptly as the ramp used in the example of *Figure 3-18*. Predicting the time at which the peaks of overshoot and undershoot occur is not as simple as with ramp excitation. A more rigorous treatment is required, including an expression for the driving waveform which closely simulates its actual shape. In the general case, a peak occurs when the sum of the slopes of the individual signal increment is zero.

## Summary

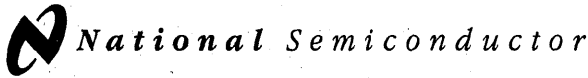
The foregoing discussions are by no means an exhaustive treatment of transmission line characteristics. Rather, they

are intended to focus attention on the general methods used to determine the interactions between high-speed logic circuits and their interconnections. Considering an interconnection in terms of distributed rather than lumped inductance and capacitance leads to the line impedance concept, i.e., mismatch between this characteristic impedance and the terminations causes reflections and ringing.

Series termination provides a means of absorbing reflections when it is likely that discontinuities and/or line impedance changes will be encountered. A disadvantage is that the incident wave is only one-half the signal swing, which limits load placement to the end of the line. ECL input capacitance increases the rise time at the end of the line, thus increasing the effective delay. With parallel termination, i.e., at the end of the line, loads can be distributed along the line. ECL input capacitance modifies the line characteristics and should be taken into account when determining line delay.

## References

1. Metzger, G. and Vabre, J., *Transmission Lines with Pulse Excitation*, Academic Press, (1969).
2. Skilling, H., *Electric Transmission Lines*, McGraw-Hill, (1951).
3. Matick, R., *Transmission Lines for Digital and Communication Networks*, McGraw-Hill, (1969).
4. Millman, J. and Taub, H., *Pulse Digital and Switching Waveforms*, McGraw-Hill, (1965).
5. "Time Domain Reflectometry", *Hewlett-Packard Journal*, Vol. 15, No. 6, (February 1964).
6. Feller, A., Kaupp H., and Digiacom, J., "Crosstalk and Reflections in High-Speed Digital Systems", *Proceedings, Fall Joint Computer Conference*, (1965).



## System Considerations

### Introduction

All of National's ECL input and output impedances are designed to accommodate various methods of driving and terminating interconnections. Controlled wiring impedance makes it possible to use simplified equivalent circuits to determine limiting conditions. Specific guidelines and recommendations are based on assumed worst-case combinations. Many of the recommendations may seem conservative, compared to typical observations, but the intent is to help the designer achieve a reliable system in a reasonable length of time with a minimum amount of redesign.

### PC Board Transmission Lines

Strictly speaking, transmission lines are not always required for F100K ECL but, when used, they provide the advantages of predictable interconnect delays as well as reflection and ringing control through impedance matching. Two common types of PC board transmission lines are microstrip and stripline, *Figure 4-1*. Stripline requires multilayer construction techniques; microstrip uses ordinary double-clad boards. Other board construction techniques are wire wrap, stitch weld and discrete wired.

Stripline, *Figure 4-1b*, is used where packing density is a high priority because increasing the interconnect layers provides short signal paths. Boards with as many as 22 layers have been used in ECL systems.

Microstrip offers easier fabrication and higher propagation velocity than stripline, but the routing for a complex system may require more design effort. In *Figure 4-1a*, the ground plane can be a part of the  $V_{EE}$  distribution as long as adequate bypassing from  $V_{EE}$  to  $V_{CC}$  (ground) is provided. Also, signal routing is simplified and an extra voltage plane is obtained by bonding two microstrip structures back to back, *Figure 4-1c*.

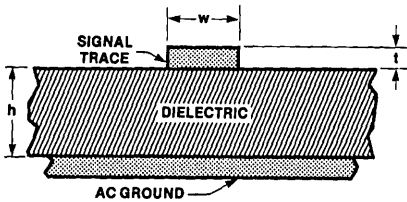
#### Microstrip

Equation 4-1 relates microstrip characteristic impedance to the dielectric constant and dimensions.<sup>1</sup> Electric field fringing requires that the ground extend beyond each edge of the signal trace by a distance no less than the trace width.

$$Z_0 = \left( \frac{60}{\sqrt{0.475 \epsilon_r + 0.67}} \right) \ln \left( \frac{4h}{0.67(0.8w + t)} \right) \quad (4-1)$$

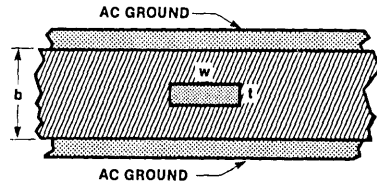
$$= \left( \frac{87}{\sqrt{\epsilon_r + 1.41}} \right) \ln \left( \frac{5.98h}{0.8w + t} \right)$$

where  $h$  = dielectric thickness,  $w$  = trace width,  $t$  = trace thickness,  $\epsilon_r$  = board material dielectric constant relative to air.



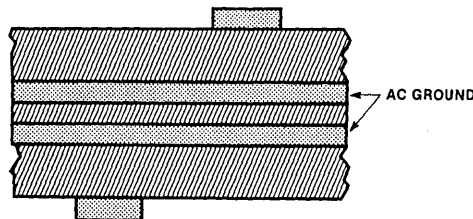
a. Microstrip

TL/F/9901-1



b. Stripline

TL/F/9901-2



c. Composite Microstrip

TL/F/9901-3

FIGURE 4-1. Transmission Lines on Circuit Boards

## PC Board Transmission Lines (Continued)

Equation 4-1 was developed from the impedance formula for a wire over ground plane transmission line, Equation 4-2.

$$Z_0 = \left(\frac{60}{\sqrt{\epsilon_r}}\right) \ln\left(\frac{4h}{d}\right) \quad (4-2)$$

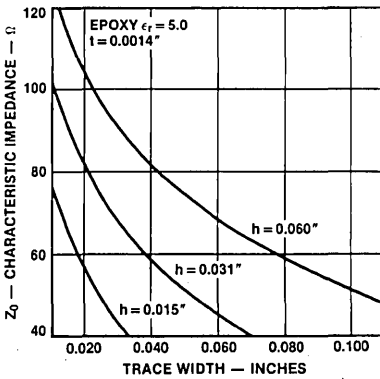
where  $d$  = wire diameter,  $h$  = distance from ground to wire center.

Comparing Equation 4-1 and 4-2, the term  $0.67(0.8w + t)$  shows the equivalence between a round wire and a rectangular conductor. The term  $0.475\epsilon_r + 0.67$  is the *effective* dielectric constant for microstrip  $\epsilon_{\text{eff}}$ , considering that a microstrip line has a compound dielectric consisting of the board material and air. The effective dielectric constant is determined by measuring propagation delay per unit of line length and using the following relationship.

$$\delta = 1.0167 \cdot \sqrt{\epsilon_{\text{eff}}} \text{ ns/ft} \quad (4-3)$$

where  $\delta$  = propagation delay, ns/ft.

Propagation delay is a property of the dielectric material rather than line width or spacing. The coefficient 1.0167 is the reciprocal of the velocity of light in free space. Propagation delay for microstrip lines on glass-filled G-10 epoxy boards is typically 1.77 ns/ft, yielding an effective dielectric constant of 3.03.

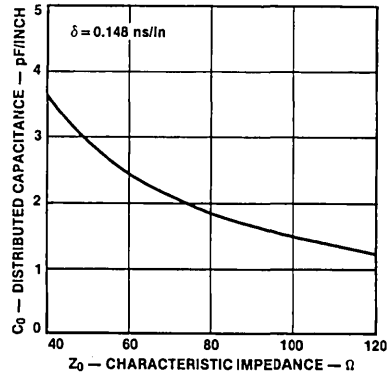


TL/F/9901-4

**FIGURE 4-2. Microstrip Impedance Versus Trace Width, G-10 Epoxy**

Using  $\epsilon_r = 5.0$  in Equation 4-1, Figure 4-2 provides microstrip line impedance as a function of width for several G-10 epoxy board thicknesses. Figure 4-3 shows the related  $C_0$  values, useful for determining capacitive loading effects on line characteristics, (Equation 3-15).

System designers should ascertain tolerances on board dimensions, dielectric constant and trace width etching in order to determine impedance variations. If conformal coating is used the effective dielectric constant of microstrip is increased, depending on the coating material and thickness.



TL/F/9901-5

**FIGURE 4-3. Microstrip Distributed Capacitance Versus Impedance, G-10 Epoxy**

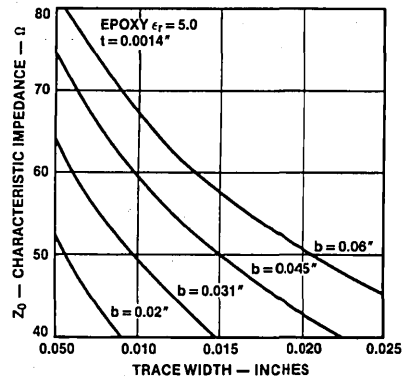
### Stripline

Stripline conductors are totally embedded. As a result, the board material determines the dielectric constant. G-10 epoxy boards have a typical propagation delay of 2.26 ns/ft. Equation 4-4 is used to calculate stripline impedances.<sup>1,2</sup>

$$Z_0 = \left(\frac{60}{\sqrt{\epsilon_r}}\right) \ln\left(\frac{4b}{0.67\pi(0.8w + t)}\right) \quad (4-4)$$

where  $b$  = distance between ground planes,  $w$  = trace width,  $t$  = trace thickness,  $w/(b-t) < 0.35$  and  $t/b < 0.25$ .

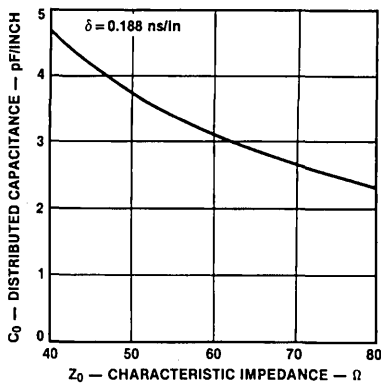
Figure 4-4 shows stripline impedance as a function of trace width, using Equation 4-4 and various ground plane separations for G-10 glass-filled epoxy boards. Related values of  $C_0$  are plotted in Figure 4-5.



TL/F/9901-6

**FIGURE 4-4. Stripline Impedance Versus Trace Width, G-10 Epoxy**

PC Board Transmission Lines (Continued)



TL/F/9901-7

FIGURE 4-5. Stripline Distributed Capacitance Versus Impedance, G-10 Epoxy

Wire Wrap

Wire-wrap boards are commercially available with five voltage planes, positions for several 24-pin Dual-In-Line Packages (DIP), terminating resistors, and decoupling capacitors. The devices are mounted on socket pins and interconnected with twisted pair wiring. One wire at each end of the twisted pair is wrapped around a signal pin, the other around a ground pin. The #30 insulated wire is uniformly twisted to provide a nominal 93Ω impedance line. Positions for Single-In-Line Package (SIP) terminating resistors are close to the inputs to provide good termination characteristics.

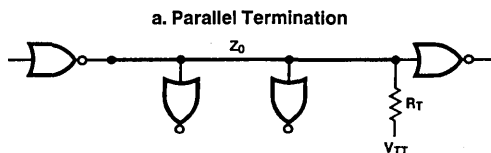
Discrete Wired

Custom Multiwire\* boards are available with integral power and ground planes. Wire is placed on a controlled thickness above the ground plane to obtain a nominal impedance line of 55Ω. Then holes are drilled through the wire and board. Copper is deposited in the drilled holes by an additive-electrolysis process which bonds each wire to the wall of the holes. Devices are soldered on the board to make connection to the wires.

\*Multiwire is a registered trademark of the Multiwire Corporation.

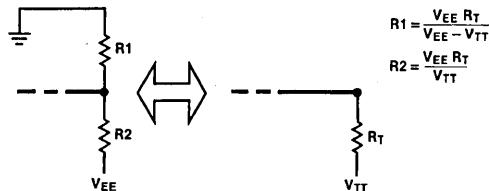
Parallel Termination

Terminating a line at the receiving end with a resistance equal to the characteristic line impedance is called parallel termination, Figure 4-6a. F100K circuits do not have internal pull-down resistors on outputs, so the terminating resistor must be returned to a voltage more negative than  $V_{OL}$  to establish the LOW-state output voltage from the emitter follower. A -2V termination return supply is commonly used. This minimizes power consumption and correlates with standard test specifications for ECL circuits. A pair of resistors connected in series between ground ( $V_{CC}$ ) and the  $V_{EE}$  supply can provide the Thevenin equivalent of a single resistor to -2V if a separate termination supply is not available, Figure 4-6b. The average power dissipation in the Thevenin equivalent resistors is about 10 times the power dissipation in the single resistor returned to -2V, as shown in Figures 5-10 and 5-13. For either parallel termination method, decoupling capacitors are required between the supply and ground (Chapter 6).



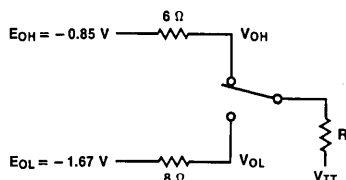
TL/F/9901-8

b. Thevenin Equivalent of  $R_T$  and  $V_{TT}$



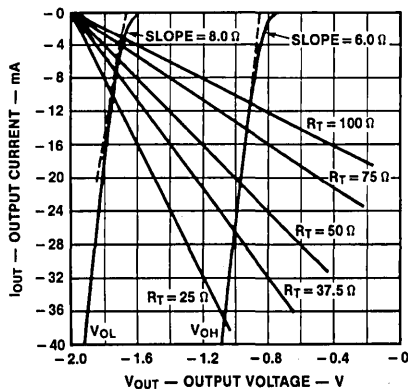
TL/F/9901-9

c. Equivalent Circuit for Determining Approximate  $V_{OH}$  and  $V_{OL}$  Levels



TL/F/9901-10

d. F100K Output Characteristic with Terminating Resistor  $R_T$  Returned to  $V_{TT} = -2.0V$



TL/F/9901-11

FIGURE 4-6. Parallel Termination

## PC Board Transmission Lines (Continued)

F100K output transistors are designed to drive low-impedance loads and have a maximum output current rating of 50 mA. The circuits are specified and tested with a 50Ω load returned to -2V. This gives nominal output levels of -0.955V at 20.9 mA and -1.705V at 5.9 mA. Output levels will be different with other load currents because of the transistor output resistance. This resistance is nonlinear with load current since it is due, in part, to the base-emitter voltage of the emitter follower, which is logarithmic with output current. With the standard 50Ω load, the effective source resistance is approximately 6Ω in the HIGH state and 8Ω in the LOW state.

The foregoing values of output voltage, output current, and output resistance are used to estimate quiescent output levels with different loads. An equivalent circuit is shown in *Figure 4-6c*. The ECL circuit is assumed to contain two internal voltage sources  $E_{OH}$  and  $E_{OL}$  with series resistances of 6Ω and 8Ω respectively. The values shown for  $E_{OH}$  and  $E_{OL}$  are -0.85V and -1.67V respectively.

The linearized portion of the F100K output characteristic can be represented by two equations:

$$\text{For } V_{OH}: V_{OUT} = -850 - 6 I_{OUT}$$

$$\text{For } V_{OL}: V_{OUT} = -1670 - 8 I_{OUT}$$

where  $I_{OUT}$  is in mA,  $V_{OUT}$  is in mV.

If the range of  $I_{OUT}$  is confined between 8 mA to 40 mA for  $V_{OH}$ , and 2 mA to 16 mA for  $V_{OL}$ , the output voltage can be estimated within ±10 mV (*Figure 4-6d*).

An ECL output can drive two or more lines in parallel, provided the maximum rated current is not exceeded. Another consideration is the effect of various loads on noise margins. For example, two parallel 75Ω terminations to -2V (*Figure 4-6d*) give output levels of approximately -1.000V and -1.716V. Noise margins are thus 35 mV less in the HIGH state and 11 mV more in the LOW state, compared to 50Ω load conditions. Conversely, a single 75Ω load to -2V causes noise margins 38 mV greater in the HIGH state and 11 mV less in the Low state, compared to a 50Ω load.

The magnitude of reflections from the terminated end of the line depends on how well the termination resistance  $R_T$  matches the line impedance  $Z_0$ . The ratio of the reflected voltage to the incident voltage  $V_i$  is the reflection coefficient  $\rho$ .

$$\frac{V_r}{V_i} = \rho = \frac{R_T - Z_0}{R_T + Z_0} \quad (4-5)$$

The initial signal swing at the termination is the sum of the incident and reflected voltages. The ratio of termination signal to incident signal is thus:

$$\frac{V_T}{V_i} = 1 + \rho = \frac{2R_T}{R_T + Z_0} \quad (4-6)$$

The degree of reflections which can be tolerated varies in different situations, but to allow for worst-case circuits, a good rule of thumb is to limit reflections to 15% to prevent excursions into the threshold region of the ECL inputs connected along the line. The range of permissible values of  $R_T$  as a function of  $Z_0$  and the reflection coefficient limitations can be determined by rearranging Equation 4-5.

$$R_T = Z_0 \frac{1 + \rho}{1 - \rho} \quad (4-7)$$

Using 15% reflection limits as examples, the range of the  $R_T/Z_0$  ratio is as follows.

$$\frac{1.15}{0.85} > \frac{R_T}{Z_0} > \frac{0.85}{1.15} \quad 1.35 > \frac{R_T}{Z_0} > 0.74 \quad (4-8)$$

The permissible range of the  $R_T/Z_0$  ratio determines the tolerance ranges for  $R_T$  and  $Z_0$ . For example, using the foregoing ratio limits,  $R_T$  tolerances of ±10% allow  $Z_0$  tolerance limits of +22% and -19%;  $R_T$  tolerances of ±5% allow  $Z_0$  tolerance limits of +28% and -23%.

An additional requirement on the maximum value of  $R_T$  is related to the value of quiescent  $I_{OH}$  current needed to insure sufficient negative-going signal swing when the ECL driver switches from the HIGH state to the LOW state. The npn emitter-follower output of the ECL circuit cannot act as a voltage source driver for negative-going transitions. When the voltage at the base of the emitter follower starts going negative as a result of an internal state change, the output current of the emitter follower starts to decrease. The transmission line responds to the decrease in current by producing a negative-going change in voltage. The ratio of the voltage change to the current change is, of course, the characteristic impedance  $Z_0$ . Since the maximum decrease in current that the line can experience is from  $I_{OH}$  to zero, the maximum negative-going transition which can be produced is the product  $I_{OH} Z_0$ .

If the  $I_{OH} Z_0$  product is greater than the normal negative-going signal swing, the emitter follower responds by limiting the current change, thereby controlling the signal swing. If, however, the  $I_{OH} Z_0$  product is too small, the emitter follower is momentarily turned off due to insufficient forward bias of its base-emitter junctions, causing a discontinuous negative-going edge such as the one shown in *Figure 4-14*. In the output-LOW state the emitter follower is essentially non-conducting for  $V_{OL}$  values more positive than about -1.55V. Using this value as a criterion and expressing  $I_{OH}$  and  $V_{OH}$  in terms of the equivalent circuit of *Figure 4-6c*, an upper limit on the value of  $R_T$  can be developed.

$$\Delta V = I_{OH} Z_0 > 1.55 - |V_{OH}|$$

$$\left( \frac{E_{OH} - V_{TT}}{R_0 + R_T} \right) Z_0 > 1.55 - \left| \frac{V_{TT} R_0 = E_{OH} R_T}{R_0 + R_T} \right|$$

$$R_T < \frac{(E_{OH} - V_{TT}) Z_0 - (1.55 - |V_{TT}|) R_0}{1.55 - |E_{OH}|} \quad (4-9)$$

For a  $V_{TT}$  of -2V,  $R_0$  of 6Ω and  $E_{OH}$  of -0.85V, Equation 4-9 reduces to

$$R_T < 1.64 Z_0 + 3.86\Omega$$

For  $Z_0 = 50\Omega$ , the emitter follower cuts off during a negative-going transition if  $R_T$  exceeds 86Ω. Changing the voltage level criteria to -1.60V to insure continuous conduction in the emitter follower gives an upper limit of 77Ω for a 50Ω line. For a line terminated at the receiving end with a resistance to -2V, a rough rule-of-thumb is that termination resistance should not exceed line impedance by more than 50%. This insures a satisfactory negative-going signal swing to ECL inputs connected along the line. The quiescent  $V_{OL}$  level, after all reflections have damped out, is determined by  $R_T$  and the ECL output characteristic.

## Input Impedance

The input impedance of ECL circuits is predominately capacitive. A single-function input has an effective value of about 2.5 pF for F100K flatpak, as determined by its effect on reflected and transmitted signals on transmission lines.

## Input Impedance (Continued)

In practical calculations, a value of 3 pF should be used. Approximately one third of this capacitance is attributed to the internal circuitry and two thirds to the flatpak pin and internal bonding.

For F100K flatpak circuits, multiple input lines may appear to have up to 4 pF to 5 pF but never more. For example, in the 100302, an input is connected internally to all five gates, but because of the philosophy of buffering these types of inputs in the F100K family this input appears as a unit load with a capacitance of approximately 2.5 pF. For applications such as a data bus, with two or more outputs connected to the same line, the capacitance of a passive-LOW output can be taken as 3 pF.

Capacitive loads connected along a transmission line increase the propagation delay of a signal along the line. The modified delay can be determined by treating the load capacitance as an increase in the intrinsic distributed capacitance of the line, discussed in Chapter 3. The intrinsic capacitance of any stubs which connect the inputs to the line should be included in the load capacitance. The intrinsic capacitance per unit length for G-10 epoxy boards is shown in *Figure 4-3* and *4-5* for microstrip and stripline respectively. For other dielectric materials, the intrinsic capacitance  $C_0$  can be determined by dividing the intrinsic delay  $\delta$  (Equation 4-3) by the line impedance  $Z_0$ .

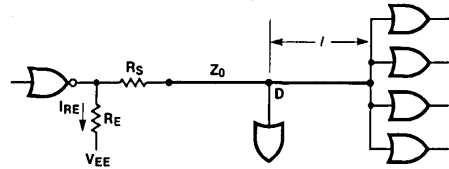
The length of a stub branching off the line to connect an input should be limited to insure that the signal continuing along the line past the stub has a continuous rise, as opposed to a rise (or fall) with several partial steps. The point where a stub branches off the line is a low impedance point. This creates a negative coefficient of reflection, which in turn reduces the amplitude of the incident wave as it continues beyond the branch point. If the stub length is short enough, however, the first reflection returning from the end of the stub adds to the attenuated incident wave while it is still rising. The sum of the attenuated incident wave and the first stub reflection provides a step-free signal, although its rise time will be longer than that of the original signal. Satisfactory signal transitions can be assured by restricting stub lengths according to the recommendations for unterminated lines (*Figure 4-10*). The same considerations apply when the termination resistance is not connected at the end of the line; a section of line continuing beyond the termination resistance should be treated as an unterminated line and its length restricted accordingly.

## Series Termination

Series termination requires a resistor between the driver and transmission line, *Figure 4-7*. The receiving end of the line has no termination resistance. The series resistor value should be selected so that when added to the driver source resistance, the total resistance equals the line impedance. The voltage divider action between the net series resistance and the line impedance causes an incident wave of half amplitude to start down the line. When the signal arrives at the unterminated end of the line, it doubles and is thus restored to a full amplitude. Any reflections returning to the source are absorbed without further reflection since the line and source impedance match. This feature, source absorption, makes series termination attractive for interconnection paths involving impedance discontinuities, such as occur in backplane wiring.

A disadvantage of series termination is that driven inputs must be near the end of the line to avoid receiving a 2-step

signal. The initial signal at the driver end is half amplitude, rising to full amplitude only after the reflection returns from the open end of the line. In *Figure 4-7*, one load is shown connected at point D, away from the line end. This input receives a full amplitude signal with a continuous edge if the distance  $l$  to the open end of the line is within recommended lengths for unterminated line (*Figure 4-10*).



TLF/9901-12

**FIGURE 4-7. Series Termination**

The signal at the end has a slower rise time than the incident wave because of capacitive loading. The increase in rise time to the 50% point effectively increases the line propagation delay, since the 50% point of the signal swing is the input signal timing reference point. This added delay as a function of the product line impedance and load capacitance is discussed in Chapter 3.

Quiescent  $V_{OH}$  and  $V_{OL}$  levels are established by resistor  $R_E$  (*Figure 4-7*), which also acts with  $V_{EE}$  to provide the negative-going drive into  $R_S$  and  $Z_0$  when the driver output goes to the LOW state. To determine the appropriate  $R_E$  value, the driver output can be treated as a simple mechanical switch which opens to initiate the negative-going swing. At this instant,  $Z_0$  acts as a linear resistor returned to  $V_{OH}$ . Thus the components form a simple circuit of  $R_E$ ,  $R_S$  and  $Z_0$  in a series, connected between  $V_{EE}$  and  $V_{OH}$ . The initial current in this series circuit must be sufficient to introduce a 0.38V transient into the line, which then doubles at the load end to give 0.75V swing.

$$I_{RE} = \frac{V_{OH} - V_{EE}}{R_E + R_S + Z_0} \geq \frac{0.38}{Z_0} \quad (4-10)$$

Any  $I_{OH}$  current flowing in the line before the switch opens helps to generate the negative swing. This current may be quite small, however, and should be ignored when calculating  $R_E$ .

Increasing the minimum signal swing into the line by 30% to 0.49V insures sufficient pull-down current to handle reflection currents caused by impedance discontinuities and load capacitance. The appropriate  $R_E$  value is determined from the following relationship.

$$\frac{V_{OH} - V_{EE}}{R_E + R_S + Z_0} \geq \frac{0.49}{Z_0} \quad (4-11)$$

For the  $R_E$  range normally used, quiescent  $V_{OH}$  averages approximately 0.955V and  $V_{EE} = -4.5V$ . The value of  $R_S$  is equal to  $Z_0$  minus  $R_0$  ( $R_0$  averages 7 $\Omega$ ). Inserting these values and rearranging Equation 4-11 gives the following.

$$R_E \leq 5.23 Z_0 + 7\Omega \quad (4-12)$$

Power dissipation in  $R_E$  is listed in *Figure 5-14*. The power dissipation in  $R_E$  is greater than in  $R_T$  of a parallel termination to  $-2V$ , but still less than the two resistors of the Thevenin equivalent parallel termination, see *Figure 5-10*, *5-13* and *5-14*.

The number of driven inputs on a series terminated line is limited by the voltage drop across  $R_S$  in the quiescent HIGH state, caused by the finite input currents of the ECL loads.  $I_{IH}$  values are specified on data sheets for various types of

## Series Termination (Continued)

inputs, with a worst-case value of 265  $\mu\text{A}$  for simple gate inputs. The voltage drop subtracts from the HIGH-state noise margin as outlined in *Figure 4-8a*.

However, there is more HIGH-state noise margin initially, because there is less  $I_{OH}$  with the  $R_E$  load than with the standard 50 $\Omega$  load to  $-2\text{V}$ . This makes  $V_{OH}$  more positive; the increase ranges from 43 mV for a 50 $\Omega$  line to 82 mV for a 100 $\Omega$  line. Using this  $V_{OH}$  increase as a limit on the voltage drop across  $R_S$  assures that the HIGH-state noise margin is as good as in the parallel terminated case. Dividing the  $V_{OH}$  increase by  $R_S + R_0 (= Z_0)$  gives the allowed load input current ( $I_x$  in *Figure 4-8a*). This works out to 0.86 mA for a 50 $\Omega$  line, 0.92 mA for a 75 $\Omega$  line and 0.82 mA for a 100 $\Omega$  line. Load input current greater than these values can be tolerated at some sacrifice in noise margin. If, for example, an additional 50 mV loss is feasible, the maximum values of current become 1.86 mA, 1.59 mA and 1.32 mA for 50 $\Omega$ , 75 $\Omega$  and 100 $\Omega$  lines respectively.

An ECL output can drive more than one series terminated line, as suggested in *Figure 4-8b*, if the maximum rated output current of 50 mA is not exceeded. Also, driving two or more lines requires a lower  $R_E$  value. This makes the quiescent  $I_{OH}$  higher and consequently  $V_{OH}$  lower, due to the voltage drop across  $R_0$ . This voltage drop decreases the HIGH-state noise margin, which may become the limiting factor (rather than the maximum rated current), depending on the particular application.

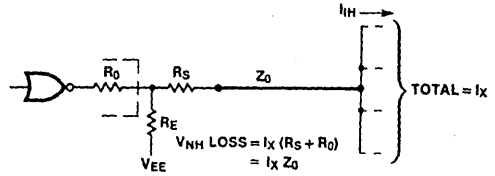
The appropriate  $R_E$  value can be determined using Equation 4-13 for  $V_{EE} = -4.5\text{V}$ .

$$\frac{1}{R_E} \geq \frac{1}{6.23 Z_1 - R_{S1}} + \frac{1}{6.23 Z_2 - R_{S2}} + \frac{1}{6.23 Z_3 - R_{S3}} \quad (4-13)$$

Circuits with multiple outputs (such as the 100313) provide an alternate means of driving several lines simultaneously (*Figure 4-8c*). Note, each output should be treated individually when assigning load distribution, line impedance, and  $R_E$  value.

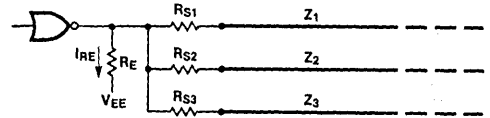
## Unterminated Lines

Lines can be used without series or parallel termination if the line delay is short compared to the signal rise time. Ringing occurs because the reflection coefficient at the open (receiving) end of the line is positive (nominally +1) while the reflection coefficient at the driving end is negative (approximately -0.8). These opposite polarity reflection coefficients cause any change in signal voltage to be reflected back and forth, with a polarity change each time the signal is reflected from the driver. Net voltage change on the line is thus a succession of increments with alternating polarity and decreasing magnitude. The algebraic sum of these increments is the observed ringing. The general relationships among rise time, line delay, overshoot and undershoot are discussed in Chapter 3, using simple waveforms for clarity. Excessive overshoot on the positive-going edge of the signal drives input transistors into saturation. Although this does not damage an ECL input, it does cause excessive recovery times and makes propagation delays unpredictable.



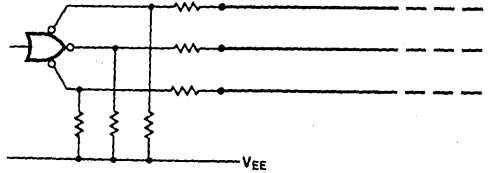
TL/F/9901-13

a. Noise Margin Loss Due to Load Input Current



TL/F/9901-14

b. Driving Several Lines from one Output



TL/F/9901-15

c. Using Multiple Output Element for Load Sharing

FIGURE 4-8. Loading Considerations for Series Termination

able. Undershoot (following the overshoot) must also be limited to prevent signal excursions into the threshold region of the loads. Such excursions could cause exaggerated transition times at the driven circuit outputs, and could also cause multiple triggering of sequential circuits. Signal swing, exclusive of ringing, is slightly greater on unterminated lines than on parallel terminated lines;  $I_{OH}$  is less and  $I_{OL}$  is greater with the  $R_E$  load, (*Figure 4-9a*) making  $V_{OH}$  higher and  $V_{OL}$  lower.

For worst case combinations of driver output and load input characteristics, a 35% overshoot limit insures that system speed is not compromised either by saturating an input on overshoot or extending into the threshold region on the following undershoot.

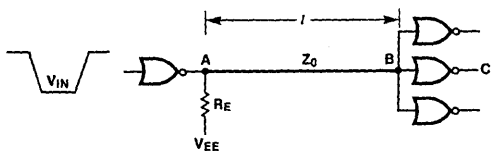
For distributed loading, ringing is satisfactorily controlled if the 2-way modified line delay does not exceed the 20% to 80% rise time of the driver output. This relationship can be expressed as follows, using the symbols from Chapter 3 and incorporating the effects of load capacitance on line delay.

$$t_r = 2T' = 2\ell\delta' = 2\ell\delta\sqrt{1 + \frac{C_L}{\ell C_0}}$$

Rearranging terms yields the quadratic equation:

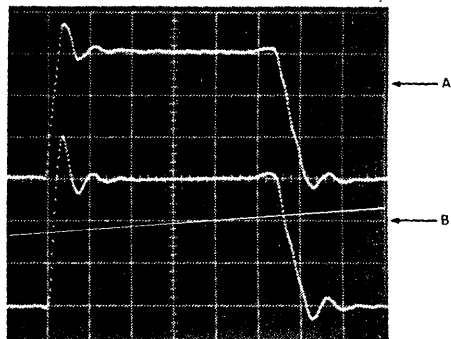
$$\ell^2 \delta_{\max}^2 + \frac{C_L}{C_0} \ell \delta_{\max} + \frac{t_r^2}{4\delta^2} = 0 \quad (4-14)$$

Unterminated Lines (Continued)



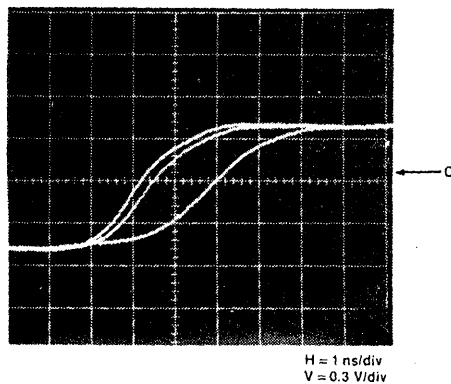
TL/F/9901-16

a. Unterminated Line



TL/F/9901-17

b. Line Voltages Showing Stair-step Trailing Edges



TL/F/9901-18

c. Load Gate Output Showing Net Propagation Increase for Increasing Values of R<sub>E</sub>: 330Ω, 510Ω, 1 kΩ

FIGURE 4-9. Effect on R<sub>E</sub> Value on Trailing-Edge Propagation

The shorter the rise time, the shorter the permissible line length. For F100K ECL, the minimum rise time from 20% to 80% is specified as 0.4 ns. Using this rise time and 3 pF per fan-out load, calculated maximum line lengths for G-10 epoxy microstrip are listed in Figure 4-10. The length (l) in the table is the distance from the terminating resistor to the input of the device(s). For other combinations of rise time, impedance, fan-out or line characteristics (δ and C<sub>0</sub>), maximum lengths are also calculated using Equation 4-14.

| Z <sub>0</sub> | Number of Fan-Out Loads |      |      |      |
|----------------|-------------------------|------|------|------|
|                | 1                       | 2    | 3    | 4    |
| 50             | 0.94*                   | 0.68 | 0.52 | 0.41 |
| 62             | 0.87                    | 0.59 | 0.44 | 0.34 |
| 75             | 0.79                    | 0.52 | 0.37 | 0.29 |
| 90             | 0.72                    | 0.45 | 0.32 | 0.25 |
| 100            | 0.68                    | 0.41 | 0.29 | 0.22 |

\*Length in inches.  
Unit load = 3 pF, δ = 0.148 ns/inch

FIGURE 4-10. F100K Maximum Worst-Case Line Lengths for Unterminated Microstrip, Distributed Loading

Resistor R<sub>E</sub> must provide the current for the negative-going signal at the driver output. Line input and output waveforms are noticeably affected if R<sub>E</sub> is too large, as shown in Figure 4-9b. The negative-going edge of the signal falls in stair-step fashion, with three distinct steps visible at point A. The waveform at point B shows a step in the middle of the negative-going swing. The effect of different R<sub>E</sub> values on the net propagation time through the line and the driven loads is evident in Figure 4-9c which shows the output signal of one driven gate in a multiple exposure photograph. The horizontal sweep (time axis) was held constant with respect to the input signal of the driver. The earliest of the three output signals occurs with an R<sub>E</sub> value of 330Ω. Changing R<sub>E</sub> to 510Ω increases the net propagation delay by 0.3 ns, the horizontal offset between the first and second signals. Changing R<sub>E</sub> to 1 kΩ produces a much greater increase in net propagation delay, indicating that the negative-going signal at B contains several steps. In practice, a satisfactory negative-going signal results when the R<sub>E</sub> value is chosen to give an initial negative-going step of 0.6V at the driving end of the line. This gives an upper limit on the value of R<sub>E</sub>, as shown in Equation 4-15.

$$\text{initial step} = \Delta l \cdot Z_0 = \frac{(V_{OH} - V_{EE}) Z_0}{R_E + Z_0} \geq 0.6$$

$$R_E \leq 5 Z_0 \text{ (at } V_{EE} = 4.5V) \quad (4-15)$$



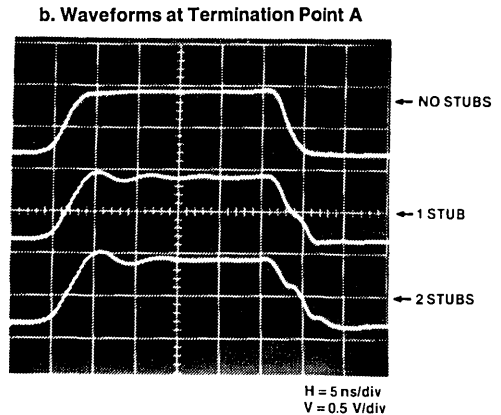
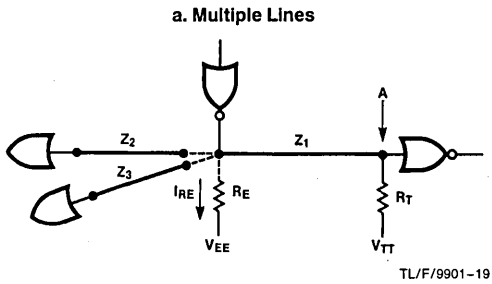
## Unterminated Lines (Continued)

An ECL output can drive two or more unterminated lines, provided each line length and loading combination is within the recommended constraints. The appropriate  $R_E$  value is determined from Equation 4-15, using the parallel impedance of the two or more lines for  $Z_0$ .

An ECL output can simultaneously drive terminated and unterminated lines, although the negative-going edge of the signal shows two or more distinct steps when the stubs are long unless some extra pull-down current is provided. *Figure 4-11a* shows an ECL circuit driving a parallel terminated line, with provision for connecting two worst-case unterminated lines to the driver output. Waveforms at the termination resistor (point A) are shown in the multiple exposure

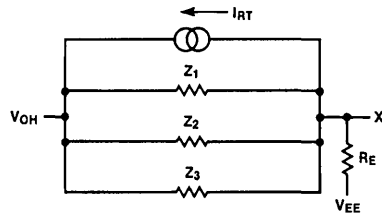
photograph of *Figure 4-11b*. The upper trace shows a normal signal without stubs connected to the driver. The middle trace shows the effect of connecting one stub to the driver. The step in the negative-going edge indicates that the quiescent  $I_{OH}$  current through  $R_T$  is not sufficient to cause a full signal for both lines. The relationship between the quiescent  $I_{OH}$  current through  $R_T$  and the negative-going signal swing was discussed earlier in connection with parallel termination.

The bottom trace in *Figure 4-11* shows the effect of connecting two stubs to the driver output. The steps in trailing edge are smaller and more pronounced. The deteriorated trailing edge of either the middle or lower waveform increases



TL/F/9901-20

**c. Equivalent Circuit for Determining Initial Negative Voltage Step at the Driver Output**



**FIGURE 4-11. Driving Terminated and Unterminated Lines in Parallel**

## Unterminated Lines (Continued)

es the switching time of the circuit connected to point A. If this extra delay cannot be tolerated, additional pull-down current must be provided. One method uses a resistor to  $V_{EE}$  as suggested in *Figure 4-11a*. The initial negative-going step at point A should be about 0.7V to insure a good fall rate through the threshold region of the driven gate. The initial step at the driver output should also be 0.7V. If the driver output is treated as a switch that opens to initiate the negative-going signal, the equivalent circuit of *Figure 4-11c* can be used to determine the initial voltage step at the driver output (point X). The value of the current source  $I_{RT}$  is the quiescent  $I_{OH}$  current through  $R_T$ . Using  $Z'$  to denote the parallel impedance of the transmission lines and  $\Delta V$  for the desired voltage step at X, the appropriate value of  $R_E$  can be determined from the following equation, using absolute values to avoid polarity confusion.

$$R_E = \left( |V_{EE}| - |V_{OH}| - \Delta V \right) \cdot \left( \frac{Z'}{|\Delta V| - |I_{RT}Z'|} \right)$$

For a sample calculation, assume that  $R_T$  and the line impedances are each 100 $\Omega$ ,  $V_{OH}$  is -0.955V,  $\Delta V$  is 0.750V,  $V_{EE}$  is -4.5V and  $V_{TT}$  is -2V.  $I_{RT}$  is thus 10.45 mA and the calculated value of  $R_E$  is 232 $\Omega$ . In practice, this value is on the conservative side and can be increased to the next larger (10%) standard value with no appreciable sacrifice in propagation through the gate at point A.

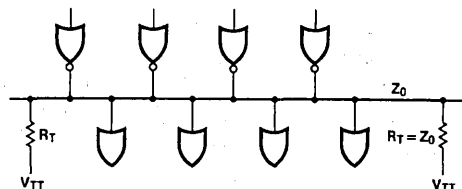
Again, the foregoing example is based on worst-case stub lengths (the longest permissible). With shorter stubs, the effects are less pronounced and a point is reached where extra pull-down current is not required because the reflection from the end of the stub arrives back at the driver while the original signal is still falling. Since the reflection is also negative going, it combines with and reinforces the falling signal at the driver, eliminating the steps. The net result is a smoothly falling signal but with increased fall time compared to the stubless condition.

The many combinations of line impedance and load make it practically impossible to define just with stub length begins to cause noticeable steps in the falling signal. A rough rule-of-thumb would be to limit the stub length to one-third of the values given in *Figure 4-10*.

## Data Bussing

Data bussing involves connecting two or more outputs and one or more inputs to the same signal line, (*Figure 4-12*). Any one of the several drivers can be enabled and can apply data to the line. Load inputs connected to the line thus receive data from the selected source. This method of steering data from place to place simplifies wiring and tends to minimize package count. Only one of the drivers can be enabled at a given time; all other driver outputs must be in the LOW or CUT-OFF state. Termination resistors matching the line impedance are connected to both ends of the line to prevent reflections. For calculating the modified delay of the line (*Chapter 3*) the capacitance of a LOW (unselected) driver output should be taken as 3 pF.

An output driving the line sees an impedance equal to half the line impedance. Similarly, the quiescent  $I_{OH}$  current is higher than with a single termination. For line impedance less than 100 $\Omega$ , the  $I_{OH}$  current is greater than the data sheet test value, with a consequent reduction of HIGH-state noise margin. This loss can be eliminated if necessary by using multiple output gates (100313) and paralleling two



TL/F/9901-22

FIGURE 4-12. Data Bus or Party Line

outputs for each driver. In the quiescent LOW state, termination current is shared among all the output transistors on the line. This sharing makes  $V_{OL}$  more positive than if only one output were conducting all of the current. For example, a 100 $\Omega$  line terminated at both ends represents a net 50 $\Omega$  DC load, which is the same as the data sheet condition for  $V_{OL}$ . If one worst-case output were conducting all the current, the  $V_{OL}$  would be -1.705V. If another output with identical DC characteristics shares the load current equally, the  $V_{OL}$  level shifts upward by about 25 mV. Connecting two additional outputs for a total of four with the same characteristics shifts  $V_{OL}$  upward another 22 mV. Connecting four more identical outputs shifts  $V_{OL}$  upward another 20 mV. Thus the  $V_{OL}$  shift for eight outputs having identical worst-case  $V_{OL}$  characteristics is approximately 67 mV. In practice, the probability of having eight circuits with worst-case  $V_{OL}$  characteristics is quite low. The output with the highest  $V_{OL}$  tends to conduct most of the current. This limits the upward shift to much less than the theoretical worst-case value. In addition, the LOW-state noise margin is specified greater than the HIGH-state margin to allow for  $V_{OL}$  shift when outputs are paralleled. Exclusive use of devices with CUT-OFF DRIVERS on the bus (i.e., 100352) will eliminate low state current sharing and preserve the low state noise margins.

In some instances a single termination is satisfactory for a data bus, provided certain conditions are fulfilled. The single termination is connected in the middle of the line. This requires that for each half of the line, from the termination to the end, the line length and loading must comply with the same restrictions as unterminated lines to limit overshoot and undershoot to acceptable levels. The termination should be connected as near as possible to the electrical mid-point of the line, in terms of the modified line delay from the termination to either end. Another restriction is that the time between successive transitions, i.e., the nominal bit time, should not be less than 15 ns. This allows time for the major reflections to damp out and limits additive reflections to a minor level.

## Wired-OR

In general-purpose wired-OR logic connections, where two or more driver outputs are expected to be in the HIGH state simultaneously, it is important to minimize the line length between the participating driver outputs, and to place the termination as close as possible to the mid-point between the two most widely separated sources. This minimizes the negative-going disturbances which occur when one HIGH output turns off while other outputs remain HIGH. The driver output going off represents a sudden decrease in line current, which in turn generates a negative-going voltage on the line. A finite time is required for the other driver outputs (quiescently HIGH) to supply the extra current. The net re-

**Wired-OR (Continued)**

sult is a "V" shaped negative glitch whose amplitude and duration depend on three factors: current that the off-going output was conducting, the line impedance, and the line length between outputs. If the separation between outputs is kept within about 1/2 inch, the transient will not propagate through the driven load circuits.

If a wired-OR connection cannot be short, it may be necessary to design the logic so that the signal on the line is not sampled for some time after the normal propagation delay (output going negative) of the element being switched. Normal propagation delay is defined as the case where the element being switched is the only one on the line in the HIGH state, resulting in the line going LOW when the element switches. In this case, the propagation delay is measured from the 50% point on the input signal of the off-going element to the 50% point of the signal at the input farthest away from the output being switched. The extra wiring time required in the case of a severe negative glitch is, in a worst-case physical arrangement, twice the line delay between the off-going output and the nearest quiescently HIGH output, plus 2 ns.

An idea of how the extra waiting time varies with physical arrangement can be obtained by qualitatively comparing the signal paths in *Figure 4-13*. With the outputs at A and B quiescently HIGH, the duration of the transient observed at C is longer if B is the off-going output than if A is the off-going element. This is because the negative-going voltage generated at B must travel to A, whereupon the corrective signal is generated, which subsequently propagates back toward C. Thus the corrective signal lags behind the initial transient, as observed at C, by twice the line delay between A and B. On the other hand, if the output at A generates the negative-going transient, the corrective response starts

when the transient reaches point B. Consequently, the transient duration observed at C is shorter by twice the line delay from A to B.

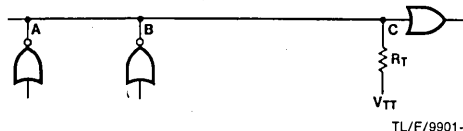
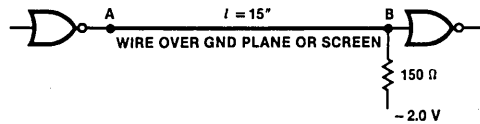


FIGURE 4-13. Relative to Wired-OR Propagation

**Backplane Interconnections**

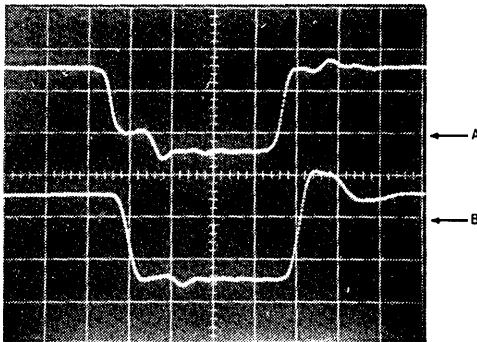
Several types of interconnections can be used to transmit a signal between logic boards. The factors to be considered when selecting a particular interconnection for a given application are cost, impedance discontinuities, predictability of propagation delay, noise environment, and bandwidth. Single-ended transmission over an ordinary wire is the most economical but has the least predictable impedance and propagation delay. At the opposite end of the scale, coaxial cable is the most costly but has the best electrical characteristics. Twisted pair and similar parallel wire interconnection cost and quality fall in between.

For single-wire transmission through the backplane, a ground plane or ground screen (Chapter 5) should be provided to establish a controlled impedance. A wire over a ground plane or screen has a typical impedance of 150Ω with variations on the order of ±33%, depending primarily on the distance from ground and the configuration of the ground. *Figure 4-14* illustrates the effects of impedance variations with a 15-inch wire parallel terminated with 150Ω to -2V. *Figure 4-14b* shows source and receiver waveforms when the wire is in contact with a continuous ground plane.



TL/F/9901-24

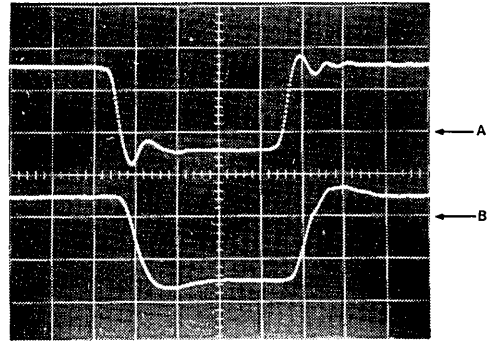
a. Wire over Ground Plane or Screen



H = 5 ns/div  
V = 0.4 ns/div

TL/F/9901-25

b. Wire in Contact with Ground Plane



H = 5 ns/div  
V = 0.4 V/div

TL/F/9901-26

c. Wire Spaced 1/8" from Ground Screen

FIGURE 4-14. Parallel Terminated Backplane Wire

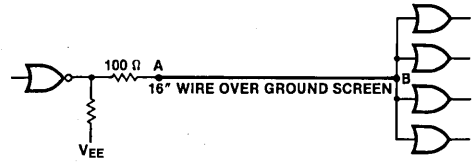
## Backplane Interconnections (Continued)

The negative-going signal at the source shows an initial step of only 80% of a full signal swing. This occurs because the quiescent HIGH-state current  $I_{OH}$  (about 7 mA) multiplied by the impedance of the wire (approximately  $90\Omega$ ) is less than the normal signal swing, and this condition allows the driver emitter follower to turn off. The negative-going signal at the receiving end is greater by 25% ( $1 + \rho = 1.25$ ). The receiving end mismatch causes a negative-going reflection which returns to the source and establishes the  $V_{OL}$  level. The positive-going signal at the source shows a normal signal swing, with the receiving end exhibiting approximately 25% overshoot.

Figure 4-14c shows waveforms for a similar arrangement, but with the wire about  $\frac{1}{8}$  inch from a ground screen. The impedance of the wire is greater than  $150\Omega$  termination, but small variations in impedance along the wire cause intermediate reflections which tend to lengthen the rise and fall times of the signal. As a result, the received signal does not exhibit pronounced changes in slope as would be expected if a  $200\Omega$  constant impedance line were terminated with  $150\Omega$ .

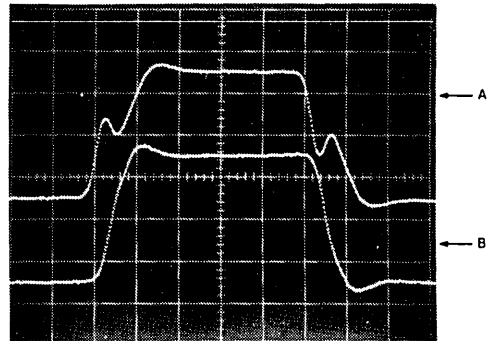
Series source resistance can also be used with single wire interconnections to absorb reflection. Figure 4-15a shows a 16-inch wire with a ground screen driven through a source resistance of  $100\Omega$ . The waveforms (Figure 4-15b) show that although reflections are generated, they are largely absorbed by the series resistor, and the signal received at the load exhibits only slight changes and overshoot. Series termination techniques can also be used when the signal into the wire comes from the PC board transmission line. Figure 4-16a illustrates a 12-inch wire over a ground screen, with 12-inch microstrip lines at either end of the wire. The output is heavily loaded (fan-out of 8) and the combination of impedances produces a variety of reflections at the input to the first microstrip line, shown in the upper trace of Figure 4-16b. The lower trace shows the final output; a comparison between the two traces shows the effectiveness of damping in maintaining an acceptable signal at the output. Figure 4-16c shows the signals at the input to the driving gate and at the output of the load gate, with a net through-put time of 8.5 ns. The circuit in Figure 4-16a is a case of mismatched transmission lines, discussed in Chapter 3.

Signal propagation along a single wire tends to be fast because the dielectric medium is mostly air. However, impedance variations along a wire cause intermediate reflections which tend to increase rise and fall times, effectively increasing propagation delay. Effective propagation delays are in the range of 1.5 to 2.0 ns per foot of wire. Load capacitance at the receiving end also increases rise and fall time (Chapter 3), further increasing the effective propagation delay.



TL/F/9901-27

a. Wire over Ground Screen



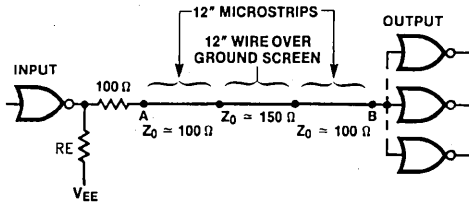
H = 10 ns/div  
V = 0.3 V/div

TL/F/9901-28

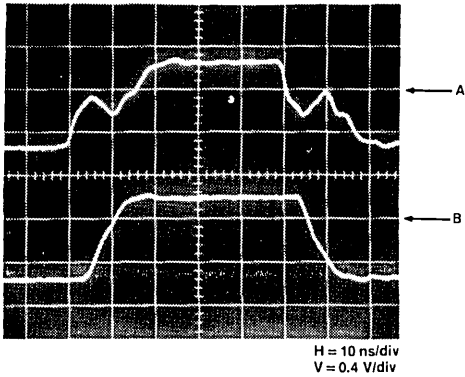
b. Series Terminated Waveform

FIGURE 4-15. Series Terminated Backplane Wire

Backplane Interconnections (Continued)

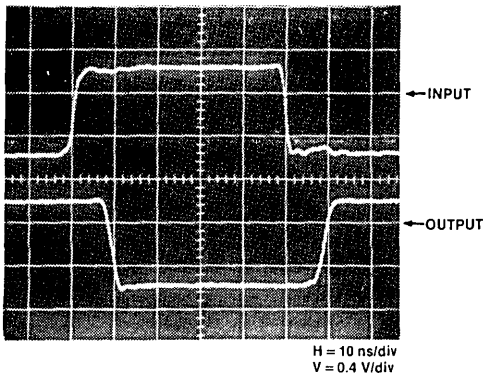


a. Backplane Wire Interconnecting PC Board Lines



TL/F/9901-30

b. Signals into the First Microstrip and at the Loads



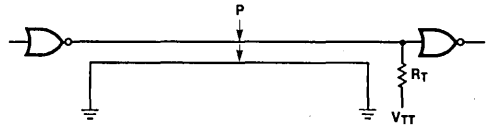
TL/F/9901-31

c. Input to Driving Gate and Output of Load Gate

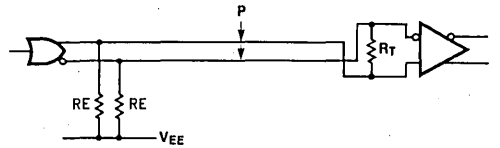
FIGURE 4-16. Signal Path with Sequence of Microstrip, Wire, Microstrip

Better control of line impedance and faster propagation can be achieved with a twisted pair. A twisted pair of AWG 26 Teflon<sup>®</sup> insulated wires, two twists per inch, exhibits a propagation delay of 1.33 ns/ft and an impedance of 115Ω. Twisted pair lines are available in a variety of sizes, impedances and multiple-pair cables. Figure 4-17a illustrates sin-

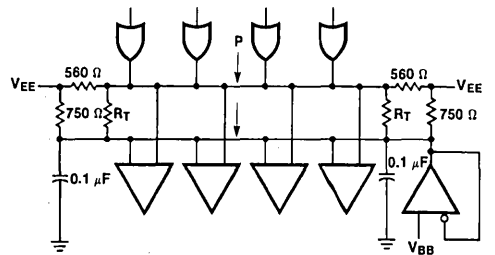
\*Teflon is a registered trademark of E.I. du Pont de Nemours Company.



a. Single-ended Twisted Pair



b. Differential Transmission Reception



c. Backplane Data Bus

FIGURE 4-17. Twisted Pair Connections

gle-ended driving and receiving. In addition to improved propagation velocity, the magnetic fields of the two conductors tend to cancel, minimizing noise coupled into adjacent wiring.

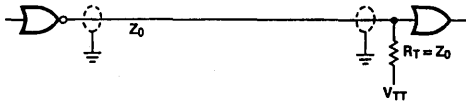
Differential line driving and receiving complementary gates as the driver and a 100314 line receiver is illustrated in Figure 4-17b. Differential operation provides high noise immunity, since common mode input voltages between -0.5V and -2.0V are rejected. The differential mode is recommended for communication between different parts of a system, because it effectively nullifies ground voltage differences. For long runs between cabinets or near high power transients, interconnections using shielded twisted pair are recommended.

Twisted pair lines can be used to implement party line type data transfer in the backplane, as indicated in Figure 4-17c. Only one driver should be enabled at a given time; the other outputs must be in the VOL state. The VBB reference voltage is available on pin 22 of the flatpak and pin 19 of the dual-in-line package for the 100314.

In the differential mode, a twisted pair can send high-frequency symmetrical signals, such as clock pulses, of 100 MHz over distances of 50 to 100 feet. For random data, however, bit rate capability is reduced by a factor of four or five due to line rise effects on time jitter.<sup>3</sup>

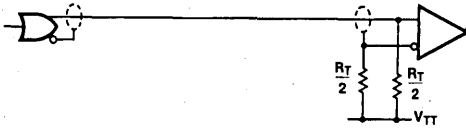
## Backplane Interconnections (Continued)

Coaxial cable offers the highest frequency capability. In addition, the outer conductor acts as a shield against noise, while the uniformity of characteristics simplifies the task of matching time delays between different parts of the system. In the single-ended mode, *Figure 4-18a*, 50 MHz signals can be transferred over distances of 100 feet. For 100 MHz operation, lengths should be 50 feet or less. In the differential mode, *Figures 4-18b, c*, the line receiver can recover smaller signals, allowing 100 MHz signals to be transferred up to 100 feet. The dual cable arrangement of *Figure 4-18c* provides maximum noise immunity. The delay of coaxial cables depends on the type of dielectric material, with typical delays of 1.52 ns/ft for polyethylene and 1.36 ns/ft for cellular polyethylene.



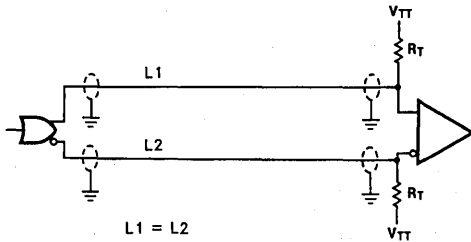
TL/F/9901-35

**a. Single-Ended Coaxial Transmission**



TL/F/9901-36

**b. Differential Coaxial Transmission**



TL/F/9901-37

**c. Differential Transmission with Grounded Shields**

**FIGURE 4-18. Coaxial Cable Connections**

## References

1. Kaupp, H. R., "Characteristics of Microstrip Transmission Lines," *IEEE Transaction on Electronic Computers*, Vol. EC-16 (April, 1967).
2. Harper, C. A., *Handbook of Wiring, Cabling and Interconnections for Electronics*. New York: McGraw-Hill, 1972.
3. True, K. M., "Transmission Line Interface Elements," *The TTL Applications Handbook*, Chapter 14 (August 1973), pp. 14-1-14-14.



Section 5  
SUPPORT MATERIALS  
**Glossary and Acronyms**



## Section 5 Contents

|                                                                       |      |
|-----------------------------------------------------------------------|------|
| Glossary of Local Area Networking and Data Communications Terms ..... | 5-3  |
| Ethernet and Networking Acronyms .....                                | 5-17 |



## Glossary of Local Area Networking and Data Communications Terms

If a term or acronym appears elsewhere in this databook and is not defined in the following glossary, please send your comments or inputs to:

Attn: LAN Applications  
National Semiconductor MS-D3635  
2900 Semiconductor Dr.  
Santa Clara, CA 95052-8090  
FAX: 408-739-6204

**AARP (Apple Address Resolution Protocol):** A protocol defined by Apple® for Appletalk® network similar in function to ARP.

**ABI (Application Binary Interface):** An application interface on AT&T's UNIX® System V Release 4. This interface enables binary compatibility for applications that run with UNIX on different platforms, and CPUs.

**Access Method:** In a data processing system, any of the techniques available to the user for moving data between main storage and an input/output device or channel. The techniques are usually part of the operating system. In Local Area Networks, the technique and/or program code used to arbitrate the use of the communications medium by granting access selectively to individual stations. Examples are CSMA/CD and Token Passing.

**ACK:** Abbreviation for acknowledgement.

**Acknowledgement:** A response sent by a receiver to indicate successful reception of information. Acknowledgements may be implemented on any networking level.

**Active Open:** The operation that a client performs to establish a TCP connection with a server at a known address.

**Adapter:** A computer add-in board. Network adapters are used to connect end-user nodes to the network; each contains an interface to a specific type of workstation or system bus. E.g.; EISA, ISA, MCA, VME, etc.

**Adaptive Routing:** A form of routing in which messages are forwarded through the network along the most cost-effective path currently available and are automatically rerouted if required by changes in the network topology (for example, if a circuit becomes disabled).

**Address:** A designator defining the ID of a DTE, peripheral device, or any other nodal component in a network. In Ethernet, each node is assigned a unique 6 byte address.

**Address Resolution:** Conversion of an Internet address into a corresponding physical address. This may require broadcasting on a local network. See ARP.

**AFP (AppleTalk File Protocol):** A network file system presentation layer protocol defined by Apple for use on Apple Macintosh® networks. It provides for access of remote file systems.

**ALAP (AppleTalk Link Access Protocol):** A Link level protocol that provides basic packet delivery transactions on Apple's Macintosh based networks.

**ANSI:** American National Standards Institute, an organization that sets information processing industry standards, represents the United States in the International Standards Organization (ISO).

**ANS X3T9.5:** The committee sponsored by ANSI which developed the standard for the Fiber Distributed Data Interface (FDDI).

**APPLETALK:** Originally defined by Apple Computer for Macintosh communication, AppleTalk protocols can now run over Ethernet networks, as well as the original, lower-speed AppleTalk network 230 Kbps (now called LocalTalk). AppleTalk is similar to NetBIOS™ in function and is supported by a number of vendors for communication between Macs and PCs, as well as simply between Macs.

AppleTalk was designed to be easy to use and requires little setup time compared to other networks.

**APPC (Advanced Program to Program Communications):** This is an interface that allows computers—and the programs running on them—to communicate over a network. APPC runs on IEEE 802.5, Ethernet, X.25 and SNA®'s synchronous data-link control. Although it was developed by IBM® and remains IBM-proprietary, work is under way within ISO to define an international protocol for "transaction programming" similar to APPC.

**Application:** A software program designed to enable end users to carry out a specific task or function. Word processors, spreadsheets, graphics programs, and database managers are examples of applications.

**Application Layer:** Level seven of the OSI model which provides the type of information transfer required, for example: file transfer or electronic messaging. See OSI.

**API (Application Program Interface):** A set of standardized interfaces to operating system functions available for use by applications programmers designed to ensure portability. Berkeley Sockets for UNIX networking and NDIS for Windows networking are examples.

**ARCnet:** A 2.5 Mbps baseband, token passing media-access protocol created by Datapoint Corp.

**ARP (Address Resolution Protocol):** Originally a Transmission Control Protocol/Internet Protocol (TCP/IP) process that maps IP addresses to Ethernet addresses; required by TCP/IP for use with Ethernet. Also referred to in other protocols as an address resolution protocol.

**ARP HACK:** See **Proxy ARP**.

**ARQ (Automatic Request for Retransmission):** A communications feature whereby the receiver asks the transmitter to resend a block or frame, generally due to errors detected by the receiver.

**ASCII (American Standard Code for Information Interchange):** A system used to represent alphanumeric data; a 7-bit-plus-parity character set established by ANSI and used for data communications and data processing; ASCII allows compatibility among data services; one of two such codes (see **EBCDIC**) used in data interchange, ASCII is normally used for asynchronous transmission.

**Asynchronous Data Transmission:** A mode of data transmission wherein the occurrence of each character is not related to a fixed time frame of reference. See **Synchronous transmission**.

**ATP (Apple Transaction Protocol):** A Transport layer protocol defined by Apple Computer. This protocol allows the reliable exchange of information between two processors on a Macintosh internet.

**AUI (Attachment-Unit Interface):** The interface between the Medium Attachment Unit (MAU) and the computing device or repeater.

**Attenuation:** The decrease in magnitude of the current, voltage or power of a signal transmitted over a wire, measured in decibels per kilometer. As attenuation increases, signal power decreases.

**Backbone Network:** High capacity network linking other networks of lower capacity. Example: FDDI as a backbone to multiple Ethernet and Token Ring LANs.

**Backoff:** In IEEE 802.3 networks when two or more nodes attempt a transmission and collide. The function of stopping transmission, and waiting a specified random time before retrying the transmission is considered backoff. In 802.3 networks a "truncated binary exponential backoff" algorithm is employed.

**BALUN (Balanced/Unbalanced):** In the IBM cabling system, refers to an impedance-matching device used to connect balanced twisted-pair cabling with unbalanced coaxial cables.

**Bandwidth:** 1. The difference, expressed in hertz, between the two limiting frequencies of a band. 2. The information capacity of a channel.

**Baseband:** A transmission scheme in which the entire bandwidth, or data-carrying capacity, of a medium (such as coaxial cable) is used to carry a single digital pulse (i.e., a signal) between multiple users. Because digital signals are not modulated, only one kind of data can be transmitted. Ethernet is among the most popular baseband LANs.

**Baud:** A unit of modulation rate or signaling speed used to designate the number of bits per second that can be transmitted in a given computer system.

**Beacon Frame:** A specialized frame in token-passing networks, sent to indicate and recover from a break in the ring.

**Big-Endian:** A binary data storage/transmission format in which the most significant bit (bit) comes first. DARPA Internet's standard is Big-Endian. See **Little-Endian**.

**BSC (BISYNC):** A family of IBM character-oriented binary synchronous communications protocols.

**Bit:** The smallest information unit in data processing. It has two possible states, "0" and "1". Bit is a contraction of Binary digit.

**Bit Duration:** The time it takes one encoded bit to pass a point on the transmission medium; in serial communications, a relative unit of time measurement, used for comparison of delay times (e.g., propagation delay, access latency) where the data rate of a (typically high-speed) transmission channel can vary.

**BER (Bit Error Rate):** The ratio of received bits that are in error relative to a specific number of bits received; measure of noise-induced distortion in digital communication links. Usually expressed as a number referenced to a power of 10; e.g., 1 error in 100,000,000,000 bits is referred to as a BER of  $1 \times 10^{-12}$ .

**Bit-Oriented:** Used to describe communications protocols in which control information may be coded in fields as small as a single bit.

**Bit Rate:** The number of bits of data transmitted over a communications line each second.

**BOOTP:** A UNIX protocol that enables diskless workstations to boot their operating system from across a network.

**BPS (Bits Per Second):** The basic unit of measure for serial data transmission capacity; Kbps for kilo (thousands of) bits per second; Mbps for mega (millions of) bits per second.

**Bridge:** Network interconnection device operating at the Media Access Control (MAC) sublayer of the OSI model's Data Link layer which provides a communication path between logically or physically separate networks. A hardware/software device that permits high-speed communication between two local or remote networks with similar or dissimilar protocols. Provides packet filtering across networks based on the Ethernet source and destination address fields. Two major bridge classifications supported are spanning tree and source routing.

**Broadband:** A means of transmission in which users are allocated different frequency channels and can therefore send data across a common path simultaneously. A data transmission scheme in which multiple signals share the bandwidth, or data-carrying capacity, of a media. This allows transmitting voice, data and video signals, for example, over a single cable, such as coaxial cable. Cable television uses broadband techniques to deliver several dozen channels over a single cable. See **Baseband**.

**Broadcast:** A method of transmitting messages to two or more stations simultaneously, such as over a bus-type local area network or by satellite; protocol mechanism whereby group and universal addressing is supported.

**Buffering:** The process of temporarily storing data in RAM, to allow transmission devices to accommodate differences in data transmission rates.

**Bus:** A length of wire or a set of parallel wires. Units which wish to intercommunicate are all connected to the bus. There must be a means of determining when each can transmit to the bus (access control), and a common method of sending and receiving the data (protocol), which includes a means of addressing to determine which unit a piece of information is for. A transmission path or channel; an electrical connection, with one or more conductors, by which all attached devices receive all transmissions simultaneously.

**Bus Topology:** The physical layout of a Local Area Network in which each node or workstation is connected directly to a length of cable or set of parallel wires.

**Byte-Oriented:** Similar to bit-oriented; control information may be coded in fields of one-byte (character) length.

**Cable Access Method:** The technique used to arbitrate the use of the communications medium by granting access selectively. (e.g., token passing and CSMA/CD).

**Cable Plant:** The physical cabling connectors, splices and patch panels in an installation.

**Caching:** A data-retrieval technique that places often-used data, such as file-allocation tables, in a computer's random-access memory or in high-speed cache memory where it can be accessed quickly.

**Carrier Sense:** The ability of each node on an Ethernet LAN to detect any traffic on the channel.

**Carrier Sense Multiple Access with Collision Detect (CSMA/CD):** The technique by which nodes on an Ethernet LAN share the transmission channel. See also Multiple Access; Carrier Sense; and Collision Detect. A LAN protocol access method for in which the nodes are attached to a cable. When a node transmits data onto the network and raises the carrier, the remaining nodes detect the carrier, Carrier Sense, and "listen" for the information to detect if it is intended for them. The nodes have network access, Multiple Access, and can send if no other transmission is taking place. If two attempt to send simultaneously a collision takes place, Collision Detection and both must retry at random intervals.

**CATV:** Cable television technology commonly employed by broadband LANs for signal distribution.

**CCITT (Consultative Committee International Telegraph and Telephony):** An international association that sets worldwide communications standard (e.g., V.21, V.22, X.25, X.25, etc.).

**Circuit Switching:** A method of communication whereby an electrical connection between calling and called stations is established on demand for exclusive use of the circuit until the connection is released.

**Channel:** A path for the transmission of information.

**Character:** Standard 8-bit unit representing a symbol, letter, number, or punctuation mark; generally means the same as byte.

**Character-Oriented:** A communications protocol or a transmission procedure that carries control information encoded in fields of one or more bytes; (compare with bit-oriented and byte-oriented).

**Character-Oriented Windows (COW) Interface:** An SAA compatible user interface for OS/2® applications.

**Characteristic Impedance:** The impedance termination of an (approximately) electrically uniform transmission line that minimizes reflections from the end of the line.

**CHEAPERNET:** Colloquial term for thin wire Ethernet, defined by IEEE 802.3 as 10Base2.

**Checksum:** The total of a group of data items or a segment of data that is used for error-checking purposes. Both numeric and alphabetic fields can be used in calculating a checksum, since the binary content of the data can be added. Just as a check digit tests the accuracy of a single number, a checksum serves to test an entire set of data which has been transmitted or stored. Checksums can detect single-bit errors and some multiple-bit errors.

**Circuit Switching:** A switching technique in which an information path (circuit) between calling and called stations is physically established on demand for exclusive use by the two stations until the connection is released. Compare to Packet Switching.

**Cluster:** Several pieces of data terminal equipment (DTE) in close proximity such that it is easy to run cabling between them.

**CMIP/CMIS:** Common Management Information Protocol (CMIP) and Common Management Information Services (CMIS) are two OSI protocols that provide a standard way of managing an OSI network.

**CMOT (CMIP/CMIS over TCP):** Use of the ISO CMIP/CMIS network management protocols to manage devices in an internet environment. See CMIP.

**Coaxial Cable:** A transmission medium with a central copper-wire conductor surrounded by concentric layers of plastic/polyvinyl chloride, aluminum or aluminized mylar and a copper tube that acts as an insulator (ground) and source of shielding from electromagnetic and radio frequency interference (EMI/RFI). Two types of coaxial cable—known as "thick" and "thin" for their respective diameters—are used in Ethernet data transmission.

**Collision Detection:** The ability of a transmitting node on an Ethernet LAN to sense a change in the energy level of the channel and to interpret the phenomenon as a collision.

**Communications Server:** A hardware/software combination that allows terminals and host computers to access a network without implementing the necessary network protocols. The communications server communicates with other devices using standard built in protocols.

**Compression:** Any of several techniques that reduce the number of bits required to represent information in data transmission or storage (thus conserving bandwidth and/or memory), in which the original form of the information can be reconstructed; also called "compaction".

**Concentrator:** Any communications device that allows a shared transmission medium to accommodate more data sources than there are channels currently available within the transmission medium.

**Connectionless Service:** The packet delivery service offered by most hardware and by the Internet Protocol. It treats each packet or datagram as a separate entity that contains both the source and destination address. Connectionless service can lose packets or deliver them out of order.

**Conditioning:** Extra-cost options that users may apply to leased, or dedicated, voice-grade telephone lines in which line impedances are carefully balanced; will generally allow for higher-quality and/or higher-speed data transmission; in increasing order of resultant line quality and cost, conditioning may be C1, C2, C4, or D1; allows improved line performance with regard to frequency response and delay distortion.

**Contention:** In communications, the situation when multiple users vie for access to a transmission channel, whether a PBX circuit, a computer port, or a time slot, within a multiplexed digital facility.

**Core:** The central region of an optical waveguide through which light is transmitted; typically 8 to 12 microns in diameter for single mode fiber, and 50 to 100 microns for multimode fiber.

**Core Gateway:** One of a set of Internet gateways which exchange routing updates periodically to ensure consistency in routing because all groups must advertise their network paths to core gateways using Exterior Gateway Protocols (EGP).

**CRC (Cyclic Redundancy Check):** A basic error-checking mechanism for link-level data transmission; a characteristic link-level feature of (typically) bit-oriented data communications protocols. The data integrity of a received frame or packet is checked via a polynomial algorithm based on the content of the frame and then matched with the result that is performed by the sender and included in a (most often, 16-bit) field appended to the frame.

**CSMA/CD (Carrier Sense Multiple Access with Collision Detection):** See Carrier Sense Multiple Access with Collision Detect (CSMA/CD).

**CSNET (Computer Science Network):** A network providing Internet connections and mail delivery service using dial-up. CSNET also provides an Internet domain name server for members who cannot run their own. CSNET was originally funded by the National Science Foundation, but is now self sufficient.

**CSU (Channel Service Unit):** A component of customer premises equipment used to terminate a digital circuit (such as DDS or T1) at the customer site; performs certain line-conditioning functions, ensures network compliance with FCC rules, and responds to loopback commands from the central office, and ensures proper "ones" density in transmitted bit stream and corrects bipolar violations. See **DSU**.

**CTI (Coax Transceiver Interface):** Ethernet coaxial cable driver/receiver which interfaces the code electronics to the physical medium. National's DP8392.

**D4 Framing:** A T1 12-frame format in which the 193rd bit is used for framing and signaling information; ESF is an equivalent but newer 24-frame technology.

**DARPA (Defense Advanced Research Projects Agency):** Formerly ARPA. A government agency that funded ARPANET and later, the DARPA Internet.

**DARPA Internet:** The collection of gateways and networks, including ARPANET, MILNET, and NSFnet, that use the TCP/IP protocol suite and operate as a single, virtual network providing reliable full duplex stream delivery and unreliable connectionless packet delivery. It also features universal connectivity and applications level services such as electronic mail.

**DAS (Dual Attach Station):** A device attached to both rings of an FDDI network.

**Data Communications:** The transmission, reception, and validation of data; data transfer between data source (origin node) and data link (destination node) via one or more data links according to appropriate protocols.

**Datagram:** A packet that includes a complete destination address along with the data it carries. A finite-length packet with sufficient information to be independently routed from source to destination. Datagram transmission typically does not involve end-to-end session establishment and may or may not entail delivery confirmation acknowledgement.

**Data Link:** 1. The physical means of connecting one location to another for the purpose of transmitting and receiving data. 2. Synonymous with communication link. Any serial data-communications transmission path, generally between two adjacent nodes or devices and without intermediate switching nodes.

**Data Link Layer:** Second layer in the OSI model; the network processing entity that establishes, maintains, and releases data-link connections between (adjacent) elements in a network to enable transmission over the physical link. See **OSI**.

**Data Terminal Equipment (DTE):** The equipment that serves as a message source or a message destination and provides for the communication control function; subscriber equipment.

**Data Transfer Rate:** The average number of bits, characters, or blocks per unit of time transferred from a data source to a data link.

**DCE (Data Circuit Terminating Equipment):** Devices that provide the functions required to establish, maintain, and terminate a data transmission connection; e.g., a modem.

**DDCMP (Digital Data Communication Message Protocol):** Digital Equipment Corporation's link level protocol. It uses serial lines, delimits frames with special characters and includes link level checksums. NSFnet incorporates DDCMP over its backbone lines.

**DDN (Defense Department Network):** MILNET and associated parts of the DARPA Internet which connect to military installations. DDN provides both local and long-haul data communications and interconnectivity for the Department of Defense systems and follows the DoD protocol suite. DDN is sometimes used to refer to MILNET, ARPANET and the TCP/IP protocols that they use.

**DDS (Dataphone Digital Service):** A private-line digital service offered undefined by BOCs and interLATA by AT&T Communications, with data rates typically at 2.4, 4.8, 9.6, and 56 Kbps; part of the services listed by AT&T under the Accunet family.

**DDS-SC: Dataphone® Digital Service with Secondary Channel;** also referred to as DDS II. A tariffed private-line service offered by AT&T and certain BOCs that allows 64 Kbps clear-channel data with a secondary channel that provides end-to-end supervisory, diagnostic, and control functions.

**DECnet®:** Digital Equipment Corporation's proprietary network architecture developed for use in WAN and includes significant Ethernet LAN capabilities, endowed with a peer-to-peer methodology.

**Dedicated Line:** A dedicated circuit, a nonswitched channel; also called a private line. See **Leased line**.

**Delay:** In communications, the time between two events; See **Propagation delay, response time**.

**Demultiplexor:** A hardware device which separates a single signal from a transmission line into several signals based on time or carrier frequency. It is used on broadband systems in combination with a multiplexor to allow multiple, simultaneous signal transmissions over a single medium. It allows multiple hardware devices to use a single communication link at the same time.

**DES (Data Encryption Standard):** A scheme approved by the National Bureau of Standards that encrypts data for security purposes. DES is the data-communications encryption standard specified by Federal Information Processing Systems (FIPS) Publication 46.

**Destination Field:** A field in a message header that contains the address of the station to which a message is being directed.

**Destination Node:** A network node to which a message is addressed.

**Disk/File Server:** A mass storage device that can be accessed by several computers; enables the creation, storage, and sharing of files.

**Disk Server:** A network device which usually gives dedicated non-shared space on a disk drive to client hosts.

**Distributed File Server:** A system by which file systems on disks distributed throughout a network are made available to workstations distributed throughout the network.

**Distributed Processing:** Processing done in multiple, separate, networked computing systems (in contrast to centralized processing where mainframes do the processing). It can be an efficient use of processing power since each CPU can be devoted to a certain task. LANs are useful in linking workstations and computer-servers that perform distributed processing.

**Distribution Frame:** A wall-mounted structure for terminating telephone wiring, usually the permanent wires from or at the telephone central office, where cross connections are readily made to extensions; also called distribution block.

**DLC (Data Link Control):** The set of rules (protocol) used by two nodes, or stations on a network to perform an orderly exchange of information. A data link includes the physical transmission medium, the protocol and associated devices and programs so it is both a physical and a logical link.

**DMA (Direct Memory Access):** A technique for high speed data transfer between a device and computer memory.

**DNA (Digital Network Architecture):** Digital Equipment Corporation's eight layer data communications protocol.

**DNIC (Data Network Identification Code):** A four digit number assigned to public data networks and to specific services within those networks.

**Domain:** A part of the Internet naming hierarchy consisting of a series of names separated by periods. For example: a host named bar.vax.edu, bar is in domain vax, vax is in domain edu.

**DOS (Disk Operating System):** The operating system utilized by most IBM and compatible personal computers.

**Dotted Decimal Notation:** The method of representing a 32-bit number with four 8-bit numbers written in base ten and separated by periods. For example 255.128.52.1.

**Driver:** See **Network Device Driver**.

**Drop Cable:** The cable that allows connection to and access from the distribution and trunk cables on an Ethernet network. Also called a transceiver cable because it runs from the network node to a transceiver (i.e., a transmitter/receiver) attached to the trunk cable.

**DSU (Data Service Unit):** A component of customer premises equipment used to interface to a digital circuit (say, DDS or T1) combined with a channel service unit (CSU) converts a customer's data stream to bipolar format for transmission.

**DTE (Data Terminal Equipment):** The equipment that serves as a message source or a message destination and provides for the communication control function; subscriber equipment.

**EBCDIC (Extended Binary Coded Decimal Interchange Code):** An 8-bit character code used primarily in IBM equipment; the code provides for 256 different bit patterns; compare with ASCII.

**EGP (Exterior Gateway Protocol):** The protocol between external gateways of autonomous systems to advertise the Internet addresses of their respective systems. Every autonomous system must use EGP to advertise network reachability to the core gateway system.

**Encryption:** The process of systematically altering or encoding data to prevent unauthorized access.

**ENDEC:** Short for Encoder/Decoder. A functional block within network adapters, that performs two basic functions. First, this function encodes the data from the controller to be transmitted over the network. Second, it decodes the data on the network to a form suitable for the network controller chip. In the case of Ethernet, this function converts NRZ controller data to Manchester data (and back again).

**Ethernet:** A branching broadcast communications system for carrying digital data packets among locally distributed computing stations. A 10 Mbit/s baseband, Local Area Network that has evolved into the IEEE 802.3 specification. A data-link protocol that specifies how data is placed on and retrieved from a common transmission medium. Ethernet is used as the underlying transport vehicle by several upper-level protocols, including TCP/IP and Xerox Network System (XNS). See **IEEE 802.3**.

**EtherTalk:** Apple's nomenclature for the Ethernet protocol that allows Apple Computer connectivity at 10 Mbps. This standard replaces the earlier AppleTalk rate of 230.4 Kbps.

**FCC (Federal Communications Commission):** Board of commissioners appointed by the President under the Communications Act of 1934, with the authority to regulate all interstate telecommunications originating in the United States.

**FCS (Frame Check Sequence):** Often referred to as CRC. This is a field in a network packet that is used to check for transmission errors in a packet sent across the network. See **CRC**.

**FDDI (Fiber Distributed Data Interface):** An ANSI Standard for high speed 100 Mbps optical fiber-based LAN with dual counter-rotating rings. Incorporates token passing and supports circuit-switched voice and packetized data. Attachment device may be through SAS or DAS.

**FDM (Frequency Division Multiplexing):** A method of transmitting multiple independent signals across a single medium by assigning each a unique carrier frequency. See **Multiplexor** and **Demultiplexor**.

**FEP (Front End Processor):** A dedicated computer linked to one or more host computers or multiuser minicomputers; performs data-communications functions and serves to off-load the attached computers of network processing, in IBM SNA networks, an IBM 3704, 3705, 3725 or 3745 communications controller.

**Fiber-Optic Cable:** A transmission medium that uses glass or plastic fibers to transport data or voice signals. Information is imposed on the glass fiber via pulses (modulation) of light from a laser or light-emitting diode (LED). Its high bandwidth, 100 to 1,000 times the information-carrying capacity of copper wire, and lack of susceptibility to electromagnetic or radio frequency interference, make fiber-optic cable ideal for use in long-haul or noisy environments, and security applications.

**File:** A collection of logically related records, usually of the same type. A named increment of storage or an unstructured or user structured form of data storage.

**FAT (File-Allocation Tables):** An area of disk that acts as an index, or directory, that tells the operating system where data has been stored on the disk. A FAT substantially increases a disk system's ability to access stored information by "pointing" the operating system to the exact location of the data it is processing.

**File Server:** A specialized computer attached to a LAN that provides data-storage service to users on a Local Area Network.

**File Transfer:** The movement of files or data from one data terminal equipment to another.

**FTP (File Transfer Protocol):** The file-sharing protocol operating at layers 5 through 7 of the Open Systems Interconnection (OSI) model that governs file sharing and file transfer capabilities.

**Flag:** In communications, a bit pattern of six consecutive "1" bits (character representation is 01111110) used in many bit-oriented protocols to mark the beginning of a frame.

**Flow Control:** The procedure or technique used to regulate the flow of data between devices; prevents the loss of data once a device's buffer has reached its capacity.

**Fourth-Generation Language (4GL):** A software productivity tool that aids programmers in the design and implementation of database management systems (DBMS).

**Fragment:** Part of a packet that is transmitted on the network. Fragments are usually generated during node collisions on 802.3 networks when one node transmits part of its packet before colliding. Also, one of the pieces that results from an Internet Gateway dividing an IP datagram into smaller pieces for transmission across a network which cannot handle the original datagram size.

**Frame:** A packet transmitted over a serial line; a physical level transmission. Derived from character-oriented protocols which added start-of-frame characters and end-of-frame characters when sending packets.

**Framing:** A control procedure used with multiplexed digital channels, such as T1 carriers, whereby bits are inserted so that the receiver can identify the time slots that are allocated to each subchannel. Framing bits may also carry alarm signals indicating specific conditions.

**Full Duplex:** The type of transmission whereby a device can receive and transmit at the same time (simultaneous, bi-directional transmission).

**Gateway:** A network interconnection device through which data flows from network to network. The gateway may reformat the data as necessary and also may participate in error and flow control protocols. Used to connect LANs employing different protocols or to public data networks.

**GBPS (Giga Bits Per Second):** A measure of the rate of data transmission referring to billions of bits per second.

**GGP (Gateway to Gateway Protocol):** The protocol used by core gateways to exchange routing information. GGP uses a shortest path routing computation.

**Half Duplex:** The type of transmission whereby a device can receive or transmit, but not at the same time.

**Hardware Address:** The low level addresses used by physical networks. Each type of hardware has its own addressing scheme.

**Head-end:** The point in a LAN where the inbound signals are transferred into outbound signals. The head-end may be passive or contain amplifier or frequency translation equipment. Used in broadband LANs and CATV.

**Header:** The control information added to the beginning of a message; contains the destination address, source address, and message number.

**Heartbeat:** In IEEE 802.3 networks this is a short burst of collision signal that is transmitted from the MAU to the DTE after every packet. Also called SQE (Signal Quality Error) test.

**Hierarchical Routing:** Routing based on hierarchical addressing by dividing the routing procedure into steps based on portions of the address. A gateway will use only the network portion of the address unless it can deliver the packet, then it also uses the host portion. Subnetting is a method of adding additional levels of hierarchical routing.

**HDLC (High-level Data Link Control):** The link level protocol defined by ISO for bit-oriented, frame-delimited data communications. An Internet standard link level communication protocol. Each frame ends with a frame check sequence for error detection. It is used in X.25 networks for link access protocol. It is increasingly used by PSN interfaces to transfer frames between a host and PSN.

**Host:** Any network node that a user can access for processing power, information files, and applications. Hosts are general purpose nodes that are not designed to perform network-specific functions.

**ICMP (Internet Control Message Protocol):** A protocol use primarily in UNIX networking. This protocol handles error and control messages, and low level functions.

**Idling Signal:** A signal used to communicate that no data is being transmitted but a connection is still established. Without idling signals, a pause in transmissions could be interpreted as a lost connection.

**IEEE 802.3:** Standard set by the IEEE for CSMA/CD network protocol, that is a Physical Layer definition including specifications for cabling in addition to transmitting data and controlling cable access. See **Ethernet**.

**IEEE 802.5:** Called token ring and typically used by PCs and large computers to communicate with IBM computers. IEEE 802.5 can transmit up to 4 megabits per second, but with recent improvements can transmit up to 16 megabits per second. It runs on coaxial, twisted-pair and fiber-optic cable. Unlike IEEE 802.3, the IEEE 802.5 network has a circular rather than a linear topology. And, rather than contending for resources, computers on the network take turns sending data by passing an electrical signal, a token, from one computer to the next. A computer can transmit data on the network only with possession of the token.

**IGP (Interior Gateway Protocol):** A term applied to any protocol used to communicate routing information and reachability within an autonomous system.

**IMP (Interface Message Processor):** Former name for Packet Switched Nodes, the Packet switches used in ARPANET. See **PSN**.

**Impedance:** The resistance a wire offers to a change in current, measured in ohms, as the current runs down the length of the wire. The greater the impedance, the shorter the distance that current can be sent down the wire. Ethernet, for example, calls for using coaxial cable with a 50 $\Omega$  impedance factor, while cable television coax offers a 75 $\Omega$  impedance factor.

**Institute of Electrical and Electronic Engineers (IEEE):** The Institute of Electrical and Electronic Engineers is a publishing and standards-making body responsible for many communications standards, including the 802.X series of LAN specifications.

**Interface:** A shared boundary; physical point of demarcation between two devices, where the electrical signals, connectors, timing, and handshaking are defined. The procedures, codes, and protocols that enable two entities to interact for exchange of information.

**Internet:** A collection of interconnected packet switched networks and gateways which function as one large network by adhering to common protocols.

**Internet Address:** The 32-bit address, consisting of a domain address and local address, assigned to a host that wants to participate with the DARPA Internet using TCP/IP.

**Internet Layer:** A network protocol layer providing host-to-host delivery over an internet. This layer encapsulates messages in IP datagrams and determines delivery pathway information. It is also responsible for handling these datagrams when they are received.

**Internet Protocol (IP):** The TCP/IP Standard Protocol which defines Internet Datagram as the unit of information passed across the Internet and provides the basis for connectionless, best-effort delivery service.

**Interoperability:** The ability of many types of hardware and software to communicate and process information in a meaningful fashion.

**IP (Internet Protocol):** See **Internet Protocol**.

**IP Datagram:** The basic unit of information passed across the internet containing source and destination address along with data.

**IPG (Interpacket Gap):** In IEEE 802.3 the minimum time between the end of one packet and beginning of another. This time is 9.6  $\mu$ s.

**IPX/SPX (Internetwork Packet Exchange/Sequenced Packet Exchange):** Network protocol. Similar in concept to TCP/IP. This is the proprietary protocol used by Novell in its NetWare products. Based on XNS protocol.

**ISDN (Integrated Services Digital Network):** An integrated digital network in which the same digital switches and digital paths are used to establish connections for voice and data traffic on the same digital links.

**Inter-Networking:** The connection of two or more networks so that work nodes on both can communicate with each other.

**ISO:** International Standards Organization

**Jabber:** A condition on an Ethernet LAN network when a node transmits for longer than the specified time.

**Jam:** In IEEE 802.3, networks when a collision occurs the colliding nodes ensure that the collision is seen by the entire network by continuing to transmit for a minimum time during a collision. Called jamming.

**Jitter:** The slight movement of a transmission signal in time or phase that can introduce errors and loss of synchronization in high-speed synchronous communications.

**Jumper:** A patch cable or wire used to establish a circuit for testing, diagnostics or configuration.

**Kilobyte (kB):** Term denoting a thousand bytes, a group of adjacent and related binary digits. A kilobyte is really 1024 bytes (this is arrived at by multiplying two by itself ten times, i.e., 2<sup>10</sup>).

**LAN (Local Area Network):** A communications system linking computers together to form a network whose dimensions typically are less than five kilometers. Transmissions within a Local Area Network generally are digital, carrying data among stations at rates usually above one megabit per second. An assembly of computing resources such as microcomputers (i.e., PCs), printers, minicomputers and mainframes linked by a common transmission medium, including coaxial cable or twisted-pair wiring.

**LAN Manager:** The multiuser network operating system co-developed by Microsoft® and 3Com®. LAN Manager offers a wide range of network management and control capabilities unavailable with existing PC-based network operating systems. It runs on Microsoft's OS/2 operating system.

**LAP (Link Access Procedure):** The data-link-level protocol specified in the CCITT X.25 interface standard; original LAP has been supplemented with LAPB (LAP-Balanced) and LAPD.

**LAPB (Link Access Procedure Balanced):** LAPB is the most common data link control protocol used to interface X.25 DTEs to X.25 DCEs. X.25 also specifies LAP (Link Access Protocol, not balanced). Both protocols are full duplex, bit synchronous protocols. The unit of transmission is a frame. Frames may contain one or more X.25 packets.

**LAPD (Link Access Procedure-D):** Link-level protocol devised for ISDN connections, differing from LAPB (LAP-Balanced) in its framing sequence.

**Latency:** The time interval between when a network station seeks access to a transmission channel and when access is granted or received; equivalent to waiting time.

**Leased Line:** A dedicated circuit, typically supplied by the telephone company, that permanently interconnects two or more user locations; generally voice-grade in capacity and in range of frequencies supported; typically analog, though sometimes it refers to DDS sub-rate digital channels (2.4 Kbps to 9.6 Kbps); used for voice (2000 Series leased line) or data (3002 type); could be point-to-point or multi-point; may be enhanced with line conditioning; also, private line.

**Link Integrity Test:** A function specified by the 10BASE-T standard which provides indication of whether the cable linking the DTE to the HUB is properly connected.

**Link Layer:** Layer two of the OSI reference model; also known as the Data-Link Layer.

**Little-Endian:** A binary data storage/transmission format in which the least significant byte (bit) comes first. See **Big-Endian**.

**LLC (Logical Link Control):** A protocol developed by the IEEE 802 committee for data-link-level transmission control; the upper sublayer of the IEEE Layer 2 OSI protocol that complements the MAC protocol; IEEE standard 802.2; includes end-system addressing and error checking.

**LU 6.2:** In Systems Network Architecture, a set of protocols that provides peer-to-peer communications between applications.

**M Bit:** The More Data mark in an X.25 packet that allows the DTE or DCE to indicate a sequence of more than one packet.

**Mail Bridge:** A mail gateway which screens mail passing from one network to another for security and administrative purposes.

**Mail Explorer:** A program which accepts a piece of mail and a list of addresses, then sends a copy of the message to each listed address.

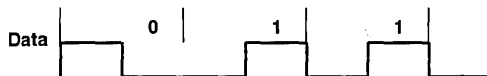
**Mail Gateway:** A machine which connects 2 or more mail systems and transfers mail among them, usually used between dissimilar systems on different networks; it will reformat messages according to destination's mailing system rules before forwarding the message.

**Mail Server:** The software and machine which provides message transfer services on a network.

**MAN (Metropolitan Area Network):** A high speed network provides facilities for data communication between sites within a neighborhood or city for distances up to 40 km and typically at data rates up to 2 Mb/s.

**MAC (Media-Access Control):** A sub-layer of the Data Link Layer, Level Two, of the ISO OSI model responsible for media control.

**Manchester Encoding:** Digital encoding technique (specified for the IEEE 802.3 Ethernet baseband network standard) in which each bit period is divided into two complementary halves; a negative-to-positive (voltage) transition in the middle of the bit period designates a binary "1", while a positive-to-negative transition represents a "0". The encoding technique also allows the receiving device to recover the transmitted clock from the incoming data stream (self-clocking).



**MAP (Manufacturing Automation Protocol):** A originated suite of networking protocols, which tracks the seven layers of the OSI model.

**Mapping:** In networking operations, the logical association of one set of values, such as addresses on one network, with quantities or values of another set, such as devices on another network (e.g., name-address mapping, internetwork route mapping, protocol-to-protocol mapping).

**MAU (Medium Attachment Unit):** The physical and electrical component that provides the means of attaching computing devices to the local network medium. In 10BASE-T networks, it is a repeater or a network interface adapter board equipped with a medium-dependent interface.

**MDI (Medium-Dependent Interface):** The mechanical and electrical interface between the twisted-pair link and the MAU. In 10BASE-T networks, the MDI is an 8-pin RJ45 modular telephone connection.

**Medium:** Any material substance used for the propagation or transmission of signals, usually in the form of electrons or modulated radio, light, or acoustic waves; such as optical fiber, cable, wire, dielectric slab, water, or air.

**MHS (Message Handling System):** The standard defined by the CCITT as X.400 and by the ISO as Message Oriented Text Interchange Standard (MOTIS).

**Mid-Level Net:** One of many networks funded by the NSF which operated autonomously but was connected to the NSFnet Backbone.

**MIFF (Minimum Internetworking Functionality):** A general principle within the ISO that calls for minimum Local Area Network station complexity when interconnecting with resources outside the Local Area Network.

**MILNET (MILitary NETWORK):** A network which was separated from ARPANET to provide reliable service to the military while ARPANET was used for continued research.

**Mini-MAP (Mini-Manufacturing Automation Protocol):** A version of MAP consisting of only physical, link, and application layers intended for lower-cost process-control networks. With Mini-MAP, a device with a token can request a response from an addressed device; unlike a standard MAP protocol, the addressed Mini-MAP device need not wait for the token to respond.

**MPR (Multi-Port Repeater):** See Repeater. A repeater with numerous network connection ports at one point on the Ethernet. In coaxial networks these repeaters typically have 8 ports; in twisted pair Ethernet up to several hundred ports are possible.

**MS<sup>®</sup> OS/2<sup>®</sup> LAN Manager:** The multiuser network operating system co-developed by Microsoft and 3Com. LAN Manager offers a wide range of network management and control capabilities unavailable with existing PC-based network operating systems.

**MTBF (Mean Time Between Failures):** A stated or published period of time for which a user may expect a device to operate before a failure occurs.

**MTTR (Mean Time To Repair):** The average time required to perform corrective maintenance on a failed device.

**Multicast Packets:** Multicast packets are addressed to a group of devices on a LAN. LAN stations use multicast packets to deliver information to a specific set of devices such as routers, servers, hosts, or nodes.

**Multimode:** An optical fiber designed to carry multiple signals, distinguished by frequency or phase, at the same time. Compare with **Single mode**.

**Multiple Access:** The ability of any node on an Ethernet LAN to send a message immediately upon sensing that the channel is free.

**Multiple Routing:** The process of sending a message to more than one recipient, usually when all destinations are specified in the header of the message.

**Multiplexor:** A hardware device which combines multiple signals from a transmission line based on time or carrier frequency. It is used on broadband systems in combination with a demultiplexor to allow multiple, simultaneous signal transmissions over a single medium. It allows multiple hardware devices to use a single communication link simultaneously.

**Multipoint Line:** A single communications line or circuit interconnecting several stations supporting terminals in several different locations. This type of line usually requires a polling mechanism, with each terminal having a unique address. Also called a multidrop line.

**Multipoint Link:** A single line that is shared by more than two nodes.

**Multitasking:** The concurrent execution of two or more tasks or applications by a computer; may also be the concurrent execution of a single program that is used by many tasks.

**MAU (Multistation Access Unit):** A multipoint connector, or concentrator, for Token Ring networks which allows devices to be connected to the ring; also provides a built-in relay that prevents a break in the network when devices are attached or removed.

**MUX:** See **Multiplexor**.



**NAK (Negative Acknowledgement):** A message sent from a receiver to a sender which indicates that the transmitted data contained errors from the transmission. Upon receiving a NAK, the sender will usually retransmit the data.

**Name Resolution:** The process of converting a host's name into a corresponding network address.

**Named Pipe:** The facility within the Microsoft OS/2 LAN Manager that allows processes on separate machines to communicate with each other across a network. Provide a simple way for application developers to write sophisticated distributed network applications.

**NCC (Network Control Center):** Any centralized network diagnostic and management station or site, such as that of a packet-switching network.

**NDIS (Network Driver Interface Specification):** Standard specification for network drivers for Microsoft's OS/2 LAN Manager.

**NetBIOS (Network Basic Input/Output System):** A programming interface to a data exchange protocol. Associated with several communications protocols and used to refer to the combination of the interface and the protocols.

It allows users and software developers to write PC programs that can communicate over a PC network in a peer-to-peer fashion.

**NetView®:** IBM's proprietary network management system that monitors, manages, and controls SNA networks. NetView allows other vendors' network management programs to communicate with it.

**NetWare™:** The popular network operating system from Novell, Inc. NetWare supports both Ethernet, Token-Ring, and FDDI network interface cards.

**Network:** An interconnected group of nodes; a series of points, nodes, or stations connected by communications channels; assembly of equipment with connections made between data stations.

**Network Address:** Refers to characters that identify the location of a node on a network. See **Address**.

**Network Architecture:** A set of design principles, including the organization of functions and the description of data formats and procedures, used as the basis for the design and implementation of a network (ISO).

**Network Interface Controller:** Electronic circuitry that connects a workstation to a network, usually a card that fits into one of the expansion slots inside a personal computer. It works with the network software and computer operating system to transmit and receive messages on the network; also, network interface card.

**Network Layer:** See **OSI**.

**NOS (Network Operating System):** The software program that provides the LAN user an interface and control of the user network interface. Network operating system communicates with the LAN hardware and the computer operating system.

**Network Management:** Administrative services performed in managing a network; e.g., network topology and software configuration, software downloading, network statistics monitoring, maintenance of network operations, and troubleshooting and diagnosis.

**Network Topology:** The physical and logical relationship of nodes in building a network configuration; the schematic arrangement of the links and nodes of a network; networks are typically a star, ring, tree, or bus topology, or hybrid combination.

**NFS (Network File System):** An extension of TCP/IP, developed by SUN Microsystems, that allows files on remote nodes of a network to appear locally connected.

**NIC (Network Interface Controller):** National's Industry-Standard 8/16 bit Ethernet Controller. Part number DP8390.

**NLM (Netware Loadable Module):** Novell's NetWare 386 supports the ability to load and run programs on the server to enhance features of the server. Programs are called NLMs.

**Node:** An endpoint of any branch of a network, or a junction common to two or more branches of a network. In a data network, a point where one or more functional units interconnect data transmission lines. Distributed system nodes include information processors, network processors, terminal controllers and terminals.

**Octet:** A data unit composed of eight ordered bits. Synonymous with byte. In FDDI, a pair of data symbols.

**ODI (Open Data Link Interface):** Standard specification created by Novell to facilitate writing of drivers under NetWare 386. Similar intent as Microsoft's NDIS.

**Off-Line:** When a user, terminal, or other device is not connected to a computer or actively transmitting via a network.

**On-Line:** Condition in which a user, terminal, or other device is actively connected with the facilities of a communications network or computer. Opposite of off-line.

**Optical Fiber:** Any filament or fiber, made of dielectric materials, that is used to transmit laser- or LED-generated light signals. Optical fiber usually consists of a core which carries the signal, and cladding, a substance with a slightly higher refractive index than the core, which surrounds the core and serves to reflect the light signal.

**OSI (Open Systems Interconnection):** A logical structure model for network operations standardized within the ISO; a seven-layer network architecture used for the definition of network protocol standards to enable any OSI-compliant computer or device to communicate with any other OSI-compliant computer or device for a meaningful exchange of information. The layers are: Physical, Data Link, Network, Transport, Session, Presentation, Application:

1. **Physical Layer**—Network wire and cable systems, defines mechanical and electrical means by which devices are physically connected to a transmission medium.
2. **Data Link Layer**—Synchronizing the flow of data and handling error control across the physical data link.
3. **Network Layer**—Provides the means to establish, maintain, and terminate connections between systems; concerned with switching and routing of information.
4. **Transport Layer**—Checks the integrity of data transported over the network.
5. **Session Layer**—Standardizes the task of setting up a session and terminating it; coordination of the interaction between stations on the network.
6. **Presentation Layer**—Defines the character set and data code, and the way data is displayed on a screen or printer format, character set, and language.
7. **Application Layer**—Links the network operating system and the application programs to perform the type of information transfer required.

**OSINET:** A test network, sponsored by the National Bureau of Standards (NBS), designed to provide a forum for doing interoperability testing for vendors of products based on the OSI model.

**Overhead:** All information, such as control, routing, and error-checking characters, in addition to user-transmitted data, including information that carries network status or operational instructions, network routing information, and re-transmissions of user data messages that are received in error.

**Out of Window Collision:** A collision on an IEEE 802.3 network that occurs outside of the specified time (for 10 Mbit/sec standards the legal collision occurs within the first 51.2  $\mu$ s of the packet).

**Packet:** A series of bits forming a complete unit of information that is sent across a network. The packet has a defined format which includes who the packet is for and who sent it (destination address source address). See **Circuit Switching**. For IEEE 802.3 the physical layer packet consists of the following fields.

|                                         |
|-----------------------------------------|
| Preamble (62 bits)                      |
| Start of Frame Delimiter (SFD) (2 bits) |
| Destination Address (6 bytes)           |
| Source Address (6 bytes)                |
| Data Field (64–1500 bytes)              |
| CRC (4 bytes)                           |

**Packet Buffer:** A memory space set aside for storing a packet that is either waiting to be transmitted, or has been received. May be located in either the network interface controller or the computer attached to the controller.

**Packet Switching:** A mode of data transmission in which messages are broken into smaller increments called packets, each routed independently to the destination. 2. The process of routing and transferring data by means of addressed packets, whereby a channel is then available for the transfer of other packets.

**PAD (Packet Assembler/Disassembler):** For X.25, a PAD allows non-X.25 users to access an X.25 network. CCITT recommendations X.3, X.28 and X.29 define PAD parameters, terminal-to-PAD interface and PAD-to-X.25 host interface.

**Pass-Through:** Describing the ability to gain access to one network element through another.

**PBX (Private Branch Exchange):** A manual, user-owned telephone exchange.

**Peer-to-Peer Communications:** The ability of intelligent computing devices to communicate without relying on a host computer.

**Physical Layer:** See **OSI**.

**PLP (Packet Level Procedures):** Defines protocols for the transfer of packets between X.25 DTE and X.25 DCE. X.25 PLP is a full duplex protocol that supports data sequencing, flow control, accountability, and error detection and recovery.

**PLS (Physical Layer Signaling):** The portion of the interface that enables MAC function communications with the AUI interface in IEEE 802.3 specifications.

**PMA (Physical Medium Attachment):** In IEEE 802.3, the portion of the MAU that contains electronic circuitry.

**Pipe:** A communications process provided by the operating system that acts as an interface between a computer's devices—keyboard, disk drives, memory, etc.—and an application program. A pipe simplifies the development of applications programs by “buffering” the application program from the intricacies of the hardware and/or the software that controls the hardware: The application developer writes “code” to a single pipe, not several individual devices. A pipe also provides communication between processes in a multitasking operating system.

**Point-to-Point Connection:** 1. In data communications, a connection established between only two data stations for data transmission. The connection may include switching facilities. Describing a circuit that interconnects two points directly, where there are generally no intermediate processing nodes, computers, or branched circuits, although there could be switching facilities; a type of connection, such as a phone line circuit, that links two, and only two, logical entities, see **multipoint line, broadcast**.

**Port:** A point of access into a computer, a network, or other electronic device; the physical or electrical interface through which one gains access; the interface between a process and a communications or transmission facility.

**Presentation Layer:** See **OSI**.

**Print Server:** An intelligent device used to transfer information to a series of printers.

**Printer Spooler:** The software that allows a user to send a file to a shared printer over a network even when the printer is busy; the file is saved in temporary storage, then printed when the printer is free.

**Promiscuous ARP:** See **Proxy ARP**.

**Protocol:** Formal set of rules governing the format, timing sequencing, and error control of exchanged messages on a data network; may be oriented toward data transfer over an interface, between two logical units directly connected, or on an end-to-end basis between two users over a large and complex network.

**Protocol Port:** A method used by transport protocols to tell the difference between many possible destinations within a single host. Operating systems usually allow an application program to specify what port it wants to be.

**Proxy ARP:** When one machine, usually a gateway, answers ARP request intended for another by supplying its own physical address, thus accepting responsibility for routing packets to the correct machine. Proxy ARP is used to allow a site to use a single Internet address with more than one physical network.

**PSDN (Packet-Switched Data Network):** A vendor-managed network that uses the X.25 protocol to transport data between customers' computers connected to the PSDN. Tariffs for PSDNs are based on the volume of data sent rather than on the distance or connect time between communicating computers.

**PSN (Packet Switched Node):** The name for an ARPANET packet switch. Each PSN is connected to at least 2 others as well as up to 16 host computers.

**Public Network:** A network operated by common carriers or telecommunications administrations for the provision of circuit-switched, packet-switched, and leased-line circuits to the public.

**Queue:** Any group of items, such as computer jobs or messages, waiting for service. In general, items are processed in "FIFO", or first-in, first-out order. In a priority queue, items of a higher priority may be serviced before lower priority items.

**Queuing:** Sequencing of batch data sessions.

**Radio Frequency (RF):** A generic term referring to the technology used in cable television and broadband Local Area Networks. Uses electromagnetic waveforms, usually in the megahertz (MHz) range, for transmission.

**RARP (Reverse Address Resolution Protocol):** The inverse of ARP. A Transmission Control Protocol/Internet Protocol (TCP/IP) process that maps Ethernet addresses back to IP addresses; required by TCP/IP for use with Ethernet. Also referred to in other protocols as a reverse address resolution protocol.

**Real Time:** Operating mode that allows immediate interaction with data as it is created, as in a process-control system or computer-aided design system.

**Redundancy:** In data transmission, the portion of a message's gross information content that can be eliminated without losing essential information; also, duplicate facilities.

**Repeater:** A device used to extend the length and topology of a physical channel, particularly a LAN cable, up to the maximum allowable end-to-end channel propagation limit.

**Response Time:** For interactive sessions, the elapsed time between the end of an inquiry and the beginning of a response.

**Retransmissive Star:** In optical-fiber transmission, a passive component that permits the light signal on an input fiber to be retransmitted on multiple output fibers; formed by heating together a bundle of fibers to near the melting point; used mainly in fiber-based Local Area Network; also, star coupler.

**RFC (Request For Comment):** A series of notes which contain information about the development of the DARPA Internet, including proposed and accepted protocols for the Internet.

**RF MODEM:** A MOdulator-DEModulator that converts digital signals to analog signals (and vice versa), then modulates/demodulates them to/from their assigned frequencies. Used in broadband LANs.

**RFS (Remote File Service):** AT&T's network file protocol for UNIX networks. Similar to NFS, except that NFS was designed for connectivity of file structures across different platforms, and RFS provides complete support for the UNIX file system semantics, and therefore can only connect UNIX file systems across the network.

**Ring:** Two or more stations connected by a physical medium wherein information is passed sequentially between active stations, each station in turn examining or copying and repeating the information, finally returning it to the originating station.

**Ring Network:** A distributed system in which the information processors are connected via a circular arrangement of data communications facilities.

**Ring Topology:** A LAN configuration in which each computer, or other node, is connected to the next, with the "last" connected to the "first" to form a complete loop, or ring. This ring may be a true physical ring or an electrical (also called logical) ring. In a physical ring, nodes are connected serially, one after the other, in a closed loop. In a logical ring, nodes are connected to a central device that routes the data-carrying signal in a circular fashion.

**RIP (Routing Information Protocol):** An interior gateway protocol (IGP) used by Berkeley BSD 4.3 UNIX systems to exchange routing information between a small number of hosts.

**RJE (Remote Job Entry):** A service offered by many networks which allows a user to submit a batch job to a host from a remote site.

**RLOGIN (Remote Login):** A service offered by Berkeley 4.3 BSD UNIX systems allowing a user on one machine to connect with other internetworked UNIX machines as if their terminal were directly attached.

**Routed (Route DAEMON):** A 4.3 BSD UNIX program which updates routing information on local area networks using RIP protocols.

**Route:** The path taken by data traffic within a network or through an internet.

**Router:** A hardware-software device that connects geographically dispersed local area networks (often, of different types, such as Ethernet and token ring) together.

**Routing:** The process of selecting the correct circuit path for a message.

**RPC (Remote Procedure Call):** A UNIX session layer protocol.

**RTT (Round Trip Time):** The time it takes for a single packet or datagram to leave one machine, reach its destination and then return to the source machine.

**RUNT Packet:** In 802.3 networks, a special case of a fragment packet when the length of the packet is less than 512 bit time.

**SAA (System Application Architecture):** An IBM developed set of standards that provides identical user interfaces for applications running on PCs, minicomputers, and mainframes.

**SAS (Single Attach Station):** A device attached to one ring of the FDDI network.

**SDLC (Synchronous Data Link Control):** A predecessor of HDLC defined by IBM Corporation and used in their SNA network products.

**Segment:** The unit of transfer between TCP's on different machines. Each segment contains a stream of bytes being sent between the machines as well as additional fields for identifications, error checking etc.

**Serial Transmission:** The sequential transmission of the bits constituting an entity of data over a data circuit.

**Server:** A specialized computer that provides a particular service, such as file or print service, to a network; increasingly, it comprises both the hardware and software which manage a network operation.

**Session Layer:** See OSI.

**SFD (Start of Frame Delimiter):** A bit pattern that enables the network board/controller to obtain byte synchronization to the incoming serial bit data from the network.

**SFT (System Fault Tolerant):** A version of Novell's NetWare that provide reliability features such as disk and file mirroring.

**Shielding:** Protective enclosure surrounding a transmission medium, such as coaxial cable, designed to minimize electromagnetic leakage and interference.

**Signal Quality Error:** See Heartbeat.

**Single Mode:** Describes an optical waveguide designed to propagate light of only a single wavelength and perhaps a single phase; essentially, an optical fiber that allows the transmission of only one light beam, or data-carrying light-wave channel, and is optimized for a particular lightwave frequency; compare with **multimode**.

**SLIP (Serial Line Internet Protocol):** Networking protocol to connect via a point-to-point serial link to network services.

**SMTP (Simple Mail Transfer Protocol):** The DARPA Internet standard protocol for transferring electronic mail messages from one machine to another. SMTP specifies how two mail systems interact and the format of control messages they exchange to transfer mail.

**SNA (Systems Network Architecture):** IBM's layered architecture for telecommunications networks. Similar in function to the OSI Reference Model.

**SNI (Serial Network Interface):** National's Manchester encoder/decoder. Part number DP8391, or CMOS version DP83910.

**SNIC (Serial Network Interface Controller):** National's newer 8/16 Controller that also incorporates the SNI encoder/decoder function in addition to the controller. It is 100% software compatible with the NIC.

**SNMP (Simple Network Management Protocol):** A protocol used to monitor the status of IP devices and the networks to which they are connected. A device which supports SNMP keeps track of status information in two data structures, the Structure of Management Information (SMI) and Management Information Block (MIB). The information is conveyed to a monitoring node by the SNMP protocol.

**Sockets:** A Berkeley BSD 4.3 network interface Specification to the transport layer. This interface provides 3 basic services. 1) Stream services which guarantees delivery of a sequence of data. 2) Datagram services which provides delivery of a data packet, but does not guarantee delivery. 3) Raw services which provide low level network functions.

**Source Node:** A network node that sends a message.

**Source Route:** A route is determined by the transmission source. The source establishes a sequence of machines that a datagram must visit to its destination.

**Spanning Tree:** A method of creating a loop-free (error free) logical topology on an extended LAN. Formation of a spanning tree topology for transmission of messages across bridges is based on the spanning tree algorithm defined in the IEEE 802.1d specification.

**SPOOL (Simultaneous Peripheral Operation On Line):** A program or piece of hardware that controls data going to an output device.

**STARLAN:** A local area network design and specification, within the IEEE 802.3 standards, characterized by 1 Mbps baseband data transmission over two-pair twisted-pair wiring.

**Star Network:** 1. A computer network in which each peripheral network node is connected only to the computer or computers at a single central facility. 2. A configuration in which remote terminals and/or processors are connected radially to a central processing location.

**STAR Topology:** A LAN configuration in which nodes are connected individually to a common device, such as a concentrator, which acts as a focal point for network cabling.

**Step-Index:** A type of optical fiber that exhibits a uniform refractive index at its core and a sharp decrease in the refractive index at its core-cladding interface.

**Store-and-Forward:** A communications technique in which messages are received at intermediate routing points and stored temporarily, then re-transmitted to an additional routing point or final destination.

**Structured Query Language (SQL):** A formal data sub-language for specifying common database operations, such as retrieving, adding, changing or deleting.

**Subnet:** A local area network which resides within another network.

**Subnet Address:** An extension of the DARPA Internet addressing scheme that allows a site to use a single internet address for many physical networks. The subnet address is not looked at by the Internet portion of the routing, it is only used by local gateways and hosts to deliver the datagram to the correct physical address.

**SYN (Synchronizing Segment):** The first segment sent by the TCP protocol, used to synchronize the two ends of a connection in preparation for opening another connection.

**Synchronous:** Communications link in which the data characters and bits are transmitted at a fixed rate with the transmitter and receiver synchronized, eliminating the need for individual start bits and stop bits surrounding each byte, thus providing greater efficiency. Contrast with asynchronous transmission.

**Synchronous Transmission:** Data transmission in which the occurrence of each signal representing a bit is related to a fixed time frame. Compare with asynchronous data transmission.

**TAP:** A device that connects a device cable to a transceiver (on baseband networks), or transfers a signal from the trunk line to a drop line (on broadband networks).

**T Carrier:** A time-division-multiplexed, typically telephone-company-supplied, digital transmission facility, usually operating at an aggregate data rate of 1.544 Mbps and above.

**TCP/IP:** The Transmission Control Protocol/Internet Protocol was formerly used only in the military, technical and university communities, but it is enjoying a surge in popularity as the commercial sector discovers its value for communicating between computers of different vendors, especially Unix systems.

TCP/IP is governed by a body of vendors and users, keeping it stable over more than a 15-year life span. TCP/IP predates OSI, and it includes several functions that properly belong in the upper level of the OSI model, such as applications that provide electronic mail, terminal emulation and file transfer.

**TELCO:** Telephone central office, in most usages; but also, a generic abbreviation for "telephone company".

**Telecommunications:** A term encompassing both voice and data communications in the form of coded signals over media.

**TELNET:** An application and protocol which allow terminal emulation over TCP/IP networks.

**Terminal:** 1. A device, such as a teletypewriter or a keyboard/CRT device, which embodies a set of human/system interface functions. 2. A point in a system or communications network at which data can either enter or leave; a device, usually equipped with a keyboard, often with a display, capable of sending and receiving data over a communications link; generically the same as data terminal equipment.

**Terminal Emulation:** A program which runs at a workstation or terminal that makes it appear to be a specific type of data terminal to both the user and the software.

**Terminal Server:** A special purpose device on an Ethernet LAN that enables up to 32 terminals to be connected to the Ethernet cable via a single physical line. A terminal server frees network nodes of the burden of establishing connections between local terminals and remote nodes. Terminals connected to the terminal server have access to all nodes on the network.

**Terminated Line:** A circuit with a resistance at the far end equal to the characteristic impedance of the line, so no reflections or standing waves are present when a signal is entered at the near end.

**Text:** In communications, transmitted characters forming the part of a message that carries information to be conveyed; in some protocols, the character sequence between start-of-text (STX) and end-of-text (ETX) control characters; information for human, as opposed to computer, comprehension, intended for presentation in a two-dimensional form.

**TFTP (Trivial File Transfer Protocol):** The DARPA Internet standard protocol for file transfer with minimal overhead and capability. It depends only on the unreliable, connectionless datagram delivery service (UDP), so it can be used on diskless workstations that keep software in ROM in order to bootstrap themselves.

**T1:** A digital transmission link with a capability of 1.544 Mbps. T1 uses two pairs of normal twisted wires to handle 24 voice channels, each one digitized at 64 Kbps. T1 is usually provided by public phone companies and used for connecting networks across remote distances.

**Timeout:** Expiration of predefined time period, at which time some specified action occurs; in communications, timeouts are employed to avoid unnecessary delays and improve traffic flow; used, for example, to specify maximum response times to polling and addressing before a procedure is automatically reinitiated.

**TPI (Twisted Pair Interface):** National's twisted pair Ethernet Transceiver for 10BASE-T. Part number DP83922.

**Token:** The password or character sequence used by network nodes to gain access to a token ring network. This character sequence (i.e., the token) passes from one node to another around the network; hence, the term "token passing" is used to describe the process.

**Token Ring:** A data-signaling network architecture where a data packet and a token are passed from one station to another along an electrical ring. When a station transmits, it takes possession of the token, transmits its data, then frees the token after the data has made a complete circuit of the electrical ring.

**TOP (Technical and Office Protocols):** A Boeing version of the MAP protocol suite aimed at office and engineering applications.

**Topology:** Description of the physical connections of a specific network's nodes—such as bus, branching bus (tree or star), or ring.

**Transaction:** In communications, a message destined for an application program; a computer-processed task that accomplishes a particular action or result; in interactive communications, an exchange between two devices, one of which is usually a computer; in batch or remote job entry, a job or job step.

**Transceiver:** A combined transmitter and receiver. An essential element of all LANs, its functions is required at each node of the network. For Ethernet it connects directly to the coaxial cable as a stand alone transceiver box. For Thin Ethernet the transceiver resides in the data terminal equipment.

**Transmission:** The dispatching of a signal, message, or other forms of intelligence by wire, radio, telegraphy, telephony, facsimile, or other means; a series of characters, messages, or blocks, including control information and user data; the signaling of data over communications channels.

**Transport Layer:** Layer four in the OSI reference model; provides a logical connection between processes on two machines. See **OSI**.

**Tree:** A LAN topology that recognizes only one route between two nodes on the network. The "map" resembles a tree or the letter T.

**Trunk:** A dedicated aggregated telephone circuit connecting two switching centers, central offices, or data-concentration devices.

**Twisted-Pair Transmission System:** In 10BASE-T terminology, refers to the twisted-pair wire link and its two attached MAUs.

**Twisted-Pair Wire:** A cable comprised of two 18 to 24 AWG (American Wire Gauge) solid copper strands twisted around each other. The twisting provides a measure of protection from electromagnetic and radio-frequency interference (EMI/RFI). Two types are available: shielded and unshielded. The former is wrapped inside a metallic sheath that provides protection from EMI/RFI. The latter, also known as telephone wire, is covered with plastic or PVC, which provides no protection from EMI/RFI.

**Type 3 Cable:** An unshielded twisted-pair wire that meets IBM specifications for use in token ring networks.

**UDP (User Datagram Protocol):** The TCP/IP transaction protocol used for applications such as remote network management and name service access; this lets users assign a name, such as "VAX™ 2", to a physical or numbered address.

**UTP (Unshielded Twisted-Pair Cable):** Also known as telephone wire. See **Twisted-Pair Wire**.

**User Transparency:** The quality in a network that enables users to access and transfer information without having to know how the network operates.

**VAN (Value Added Network):** A network whose services go beyond simple switching.

**VC (Virtual Circuit):** For X.25 a VC is a PLP logical connection between an X.25 DCE and an X.25 DTE. X.25 supports both switched VCs and permanent VCs. Switched VCs are analogous to dial up lines. They allow a particular X.25 DTE to establish connection with different X.25 DTEs on a per call basis. In contrast, permanent VCs are analogous to leased lines because they always connect two particular X.25 DTEs.

**Virtual Disk:** A portion of physical disk drive appearing to a dedicated host as a local disk resource.

**Virtual Storage:** Storage space that may be viewed as addressable main storage, but is actually auxiliary storage (usually peripheral mass storage) mapped into real addresses; amount of virtual storage is limited by the addressing scheme of the computer.

**VMST<sup>™</sup> (Virtual Memory System):** An Operating System developed by Digital Equipment Corporation for the VAX computer series.

**Well-Known Port:** Any set of protocol port numbers preassigned for specific uses by the transport layer protocols (i.e., TCP and UDP). Clients can locate servers at well-known port assignments. File transfer servers, echo servers and time servers are some examples of servers using well-known port assignments.

**WAN (Wide Area Network):** A network covering a large geographic area (50 miles or more); may include packet-switched, public data, and Value-Added Networks.

**Wide Band:** A system in which multiple channels access a medium (usually coaxial cable) that has a large bandwidth (10 Mbps is typical) using radio frequency modems. Each channel is modulated to a different frequency slot on the cable and is demodulated to its original frequency at the receiving end.

**Wiring Closet:** Central location for termination and routing on-premises wiring systems.

**Workstation:** Input/Output equipment that an operator works.

**XDR (External Data Representation):** A presentation layer protocol used by SUN microsystems. It provides a common way of representing data on a network consisting of different machines.

**XNS (Xerox Network System):** Used as the basis for many network operating systems, including early version of 3Com's 3+. It performs functions similar to those of TCP/IP and runs on top of IEEE 802.3.

**X.NN:** The X.nn series of the CCITT standards relate to the connection of digital equipment to a public data network which employs digital signaling.

**XON/XOFF (Transmitter On/Transmitter Off):** A method of flow control used when a computer is attached to a slower device which cannot process information as fast as the computer sends it. A common device using XON/XOFF is a printer. XON is sent as a CONTROL-Q, XOFF is sent as a CONTROL-S.

**X.25:** Defined by CCITT, and used most commonly in Europe.

The X.25 protocol is best suited for small to medium amounts of data traffic among multiple locations. It operates over telephone lines at up to 1.5 megabits per second (the maximum speed of a T-1 connection). X.25 breaks a data message into smaller pieces known as "packets" and transmits the packets individually to their destination, where they are reassembled. Because each packet is routed individually, packets may travel different paths to their destination, thereby speeding transmission.

**X.400:** This standard, approved by ISO, defines a means for exchanging electronic mail between computers. Supporting this standard allows electronic mail packages from different vendors to exchange messages.

**X.500:** Still under development, X.500 is a standard for directory management. It will allow users to find files and data on networks of different types of computers.

**10BASE5:** IEEE 802.3 Physical Layer Standard for thick cable Ethernet, utilizing thick double shielded coaxial cable. 10BASE5 stands for; 10 = 10 Mbits/sec. data rate, BASE = Baseband, 5 = 500 meters segment length.

**10BASE2:** IEEE 802.3 Physical Layer Standard for thin wire Ethernet (sometimes called cheapernet). This standard uses RG58 standard coaxial cable. 10BASE2 stands for; 10 = 10 Mbit/sec. data rate, BASE = Baseband, 2 = 200 meter segment length (actually is 185m).

**1BASE5:** IEEE 802.3 Physical Layer Standard for StarLAN twisted pair network. 1BASE5 stands for; 1 = 1 Mbit/sec. data rate, BASE = Baseband, 5 = 500 meter segment length.

**10BASE-T:** IEEE 802.3 Physical Layer Standard for the new twisted pair Ethernet in a star topology. 10BASE-T stands for; 10 = 10 Mbit/sec. data rate, BASE = Baseband, T = twisted pair wire over 100 meters nominal segment length.

**802.x:** The Institute of Electrical and Electronic Engineers (IEEE) committees that developed a set of standards defining some networks. The IEEE committees generally work on standards below 50 Mb/s including:

- IEEE 802.3 Ethernet
- IEEE 802.4 Token Bus
- IEEE 802.5 Token Ring
- IEEE 802.6 Metropolitan Area Networks
- IEEE 802.9 Integrated Data and Voice

**3270 and 5250:** These two IBM protocols have been around since the early 1970s. High Speed Serial Interface used with IBM mainframes and various peripherals. Data rates range from 1.2 Mbits/s to 52 Mbits/s.

## Ethernet and Networking Acronyms

| Acronym           | Description                                               |
|-------------------|-----------------------------------------------------------|
| ANSI              | American National Standards Institute                     |
| API               | Application Program Interface                             |
| AUI               | Attachment Unit Interface                                 |
| BER               | Bit Error Rate                                            |
| bps               | Bits Per Second                                           |
| BMACTM            | Media Access Controller (FDDI)<br>National's DP83261      |
| BSITM/<br>BSI-2TM | System Interface (FDDI)<br>National's DP83265/65A         |
| CDDTM             | Clock Distribution Device (FDDI)<br>National's DP83241    |
| CFM               | Configuration Management                                  |
| CMIP              | Common Management Information Protocol                    |
| CMT               | Connection Management                                     |
| CPU               | Central Processing Unit                                   |
| CRC               | Cyclic Redundancy Check                                   |
| CRD               | Clock Recovery Device—<br>Part of National's Solution     |
| CSMA/CD           | Carrier-Sense Multiple<br>Access with Collision Detection |
| CTI               | Coaxial Transceiver Interface, NSC<br>DP8392              |
| DA                | Destination Address                                       |
| DAC               | Dual Attach Concentrator                                  |
| DAS               | Dual Attach Station                                       |
| DIW               | Distributed Inside Wire                                   |
| DLL               | Data Link Layer                                           |
| DTP               | Datagrade Unshielded Twisted Pair                         |
| EIA               | Electronics Industries Association                        |
| ENDEC             | ENcoder, DECoder combination,<br>chip or function         |
| FC                | Frame Control                                             |
| FCS               | Frame Check Sequence                                      |
| FDDI              | Fiber Distributed Data Interface                          |
| FTP               | File Transfer Protocol                                    |
| IEEE              | Institute of Electrical and Electronic Engineers          |
| IP                | Internetwork Protocol                                     |
| ISDN              | Integrated Services Digital Network                       |
| ISO               | International Standards Organization                      |

| Acronym | Description                                                                                    |
|---------|------------------------------------------------------------------------------------------------|
| LAN     | Local Area Network                                                                             |
| LED     | Light Emitting Diode                                                                           |
| LEM     | Link Error Monitor                                                                             |
| LERIC   | Low End Repeater Interface Controller                                                          |
| LLC     | Logical Link Control                                                                           |
| MAC     | Media Access Control Layer                                                                     |
| MAU     | Media Access Unit                                                                              |
| MIB     | Management Information Base                                                                    |
| MPR     | Multi-Port Repeater                                                                            |
| NFS     | Network File System                                                                            |
| NIC     | Network Interface Controller (add-in<br>card); also NSC DP8390 Network<br>Interface Controller |
| NLM     | Netware Loadable Module                                                                        |
| NOS     | Network Operating System                                                                       |
| NRZ     | Non-Return to Zero                                                                             |
| NRZI    | Non-Return to Zero Invert on Ones                                                              |
| OSI     | Open Systems Interconnection                                                                   |
| PDU     | Protocol Data Unit                                                                             |
| PHY     | Physical Layer                                                                                 |
| PMD     | Physical Medium Dependent Layer                                                                |
| QLS     | Quiet Line State                                                                               |
| RIC     | Repeater Interface Controller, NSC<br>DP83950                                                  |
| SA      | Source Address                                                                                 |
| SAS     | Single Attach Station                                                                          |
| SDU     | Service Data Unit                                                                              |
| SNA     | Systems Network Architecture (IBM)<br>Protocol                                                 |
| SNMP    | Simple Network Management Protocol                                                             |
| SONIC   | Systems Oriented Network Interface<br>Controller, NSC DP83932                                  |
| ST-NIC  | Serial Network Interface Controller for<br>Twisted Pair, NSC DP83902                           |
| TCP/IP  | Transmission Control<br>Protocol/Internet Protocol                                             |
| WAN     | Wide Area Network                                                                              |
| XNS     | Xerox Network System Protocol                                                                  |







Section 6  
**Appendix/  
Physical Dimensions**

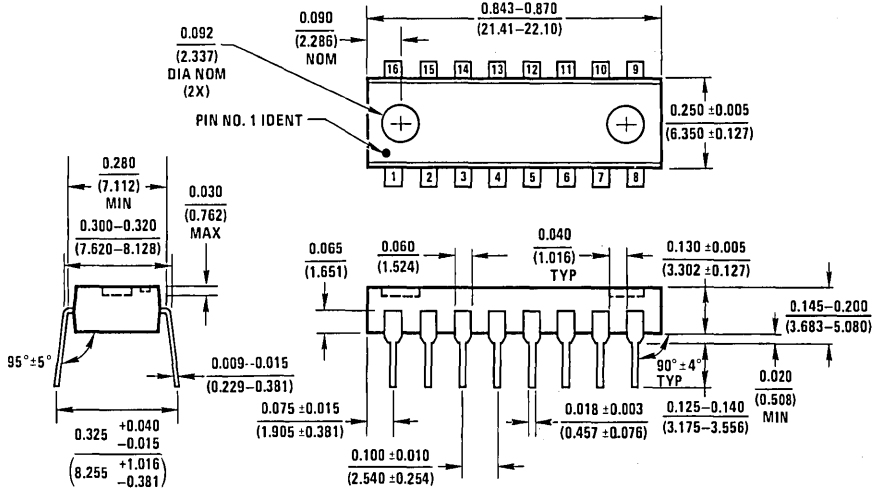


## **Section 6 Contents**

|                           |     |
|---------------------------|-----|
| Physical Dimensions ..... | 6-3 |
| Bookshelf                 |     |
| Distributors              |     |
| Worldwide Sales Offices   |     |

**16 Lead (0.300" Wide) Molded Dual-in-Line Package**  
**NS Package Number N16A**

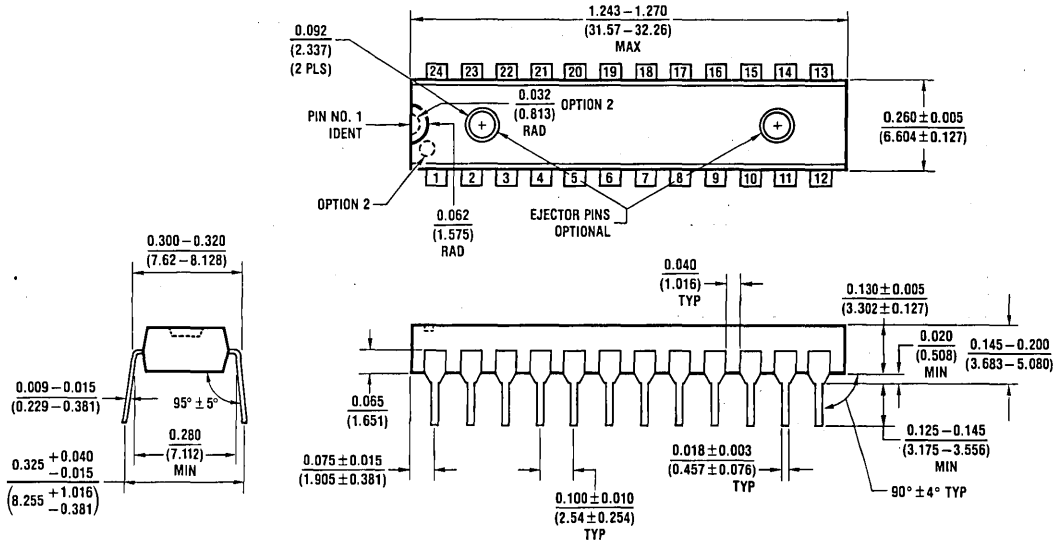
All dimensions are in inches (millimeters)



N16A (REV E)

**24 Lead (0.300" Wide) Molded Dual-in-Line Package**  
**NS Package Number N24C**

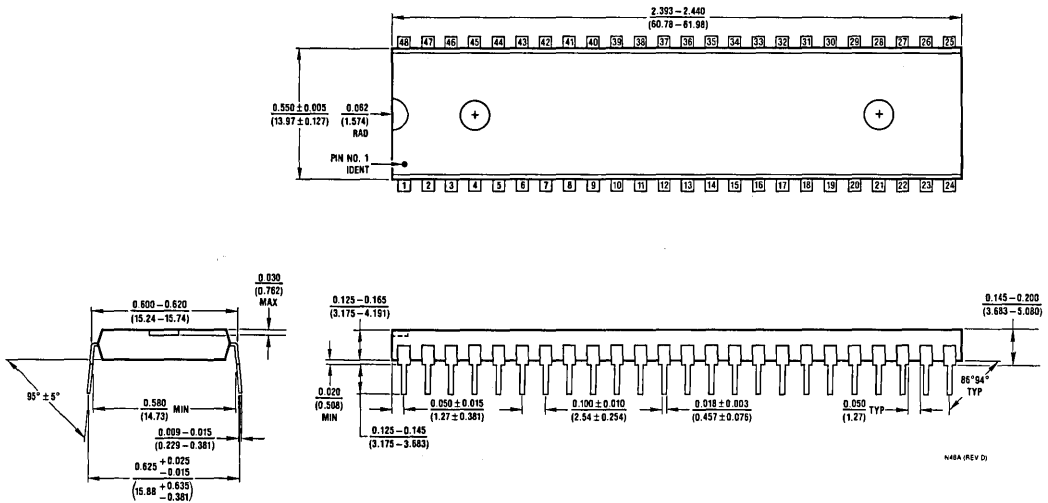
All dimensions are in inches (millimeters)



N24C (REV F)

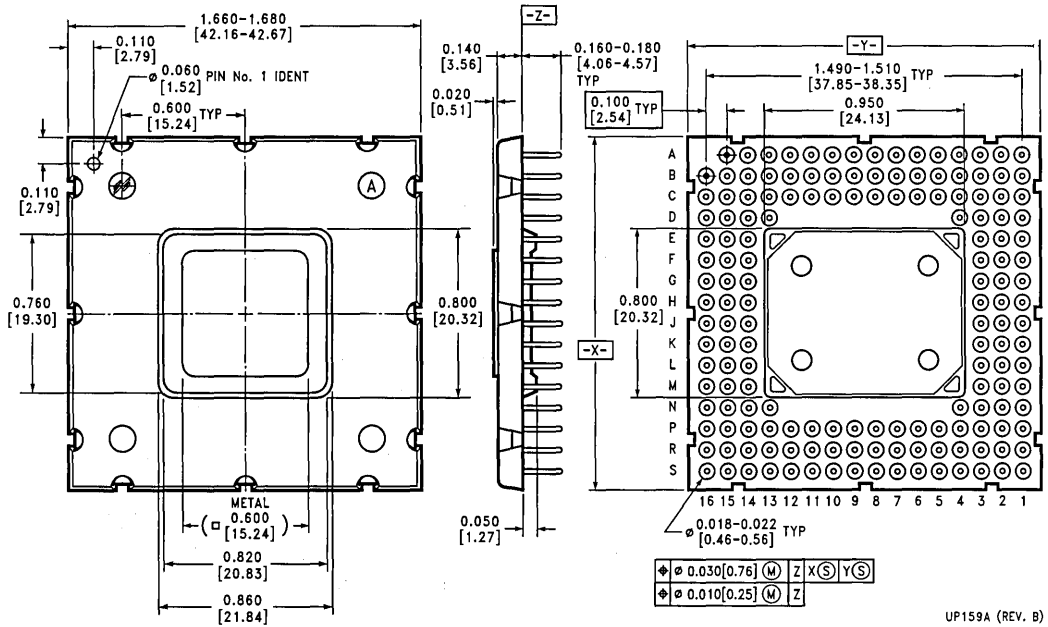
### 48 Lead (0.600" Wide) Molded Dual-in-Line Package NS Package Number N48A

All dimensions are in inches (millimeters)



### 159 Pin Molded Plastic Pin Grid Array NS Package Number UP159A

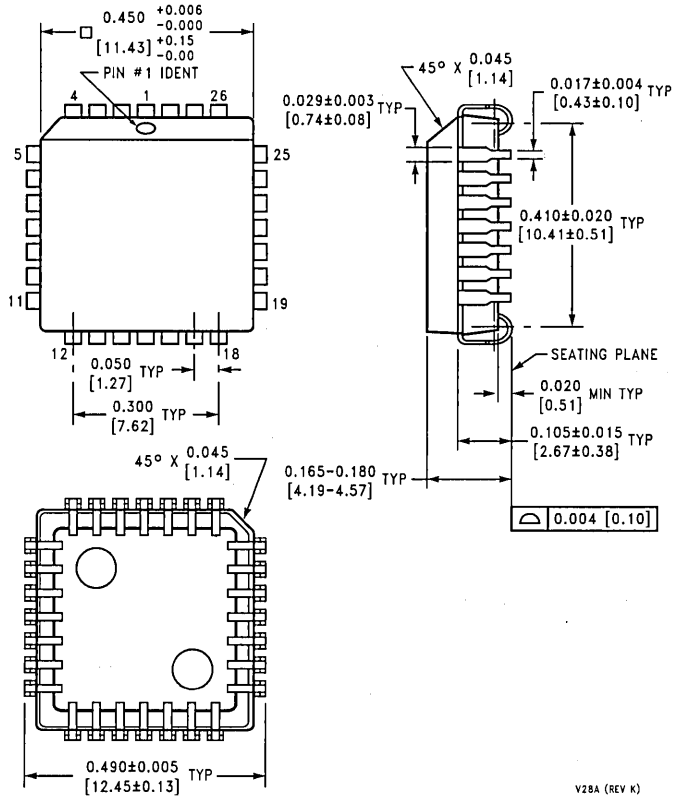
All dimensions are in inches [millimeters]



UP159A (REV. B)

# 28 Lead Molded Plastic Leaded Chip Carrier NS Package Number V28A

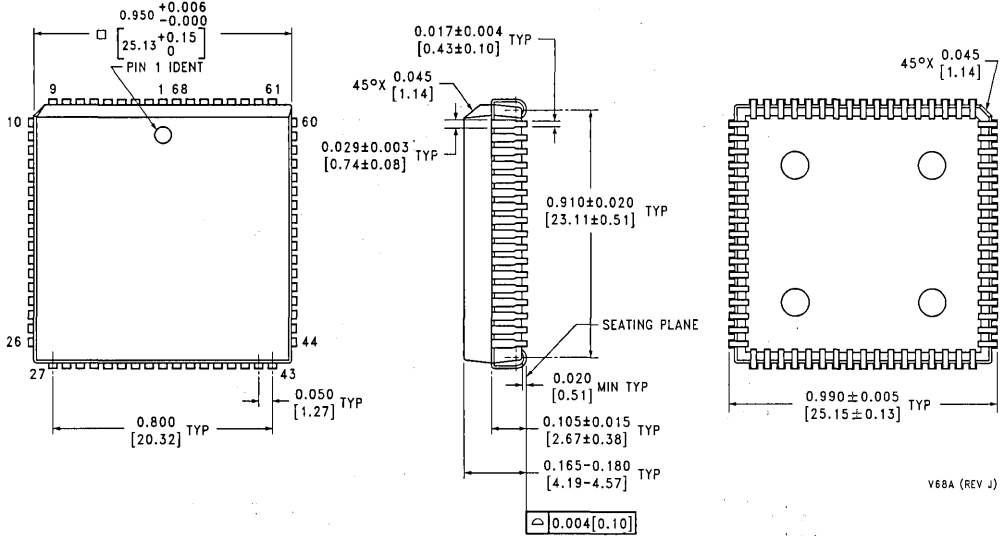
All dimensions are in inches [millimeters]



V28A (REV K)

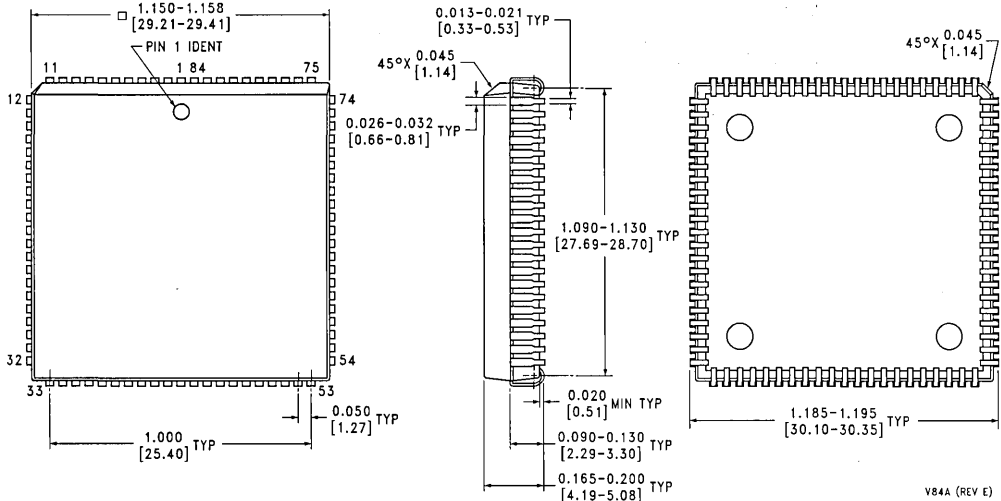
### 68 Lead Molded Plastic Leaded Chip Carrier NS Package Number V68A

All dimensions are in inches [millimeters]



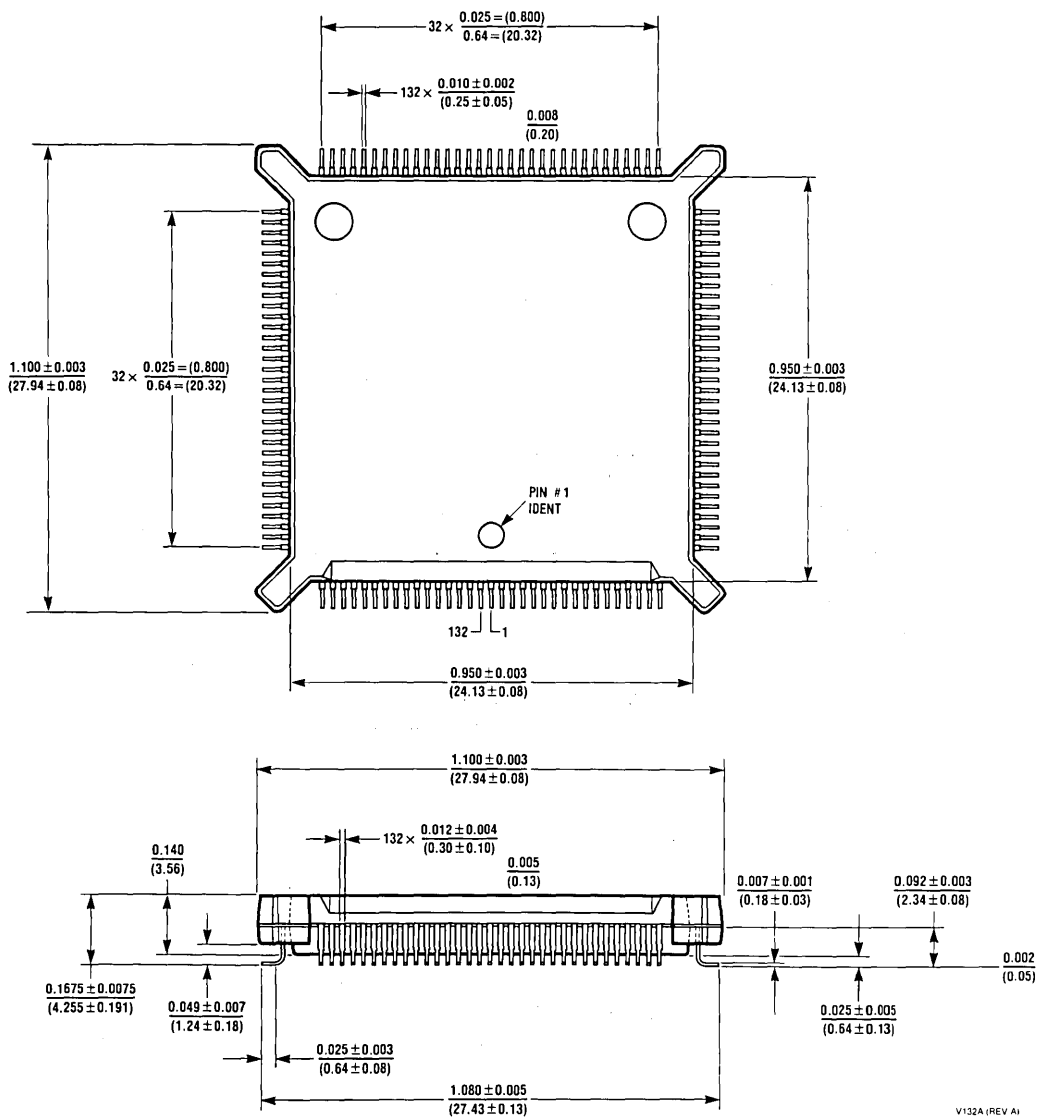
### 84 Lead Molded Plastic Leaded Chip Carrier NS Package Number V84A

All dimensions are in inches [millimeters]



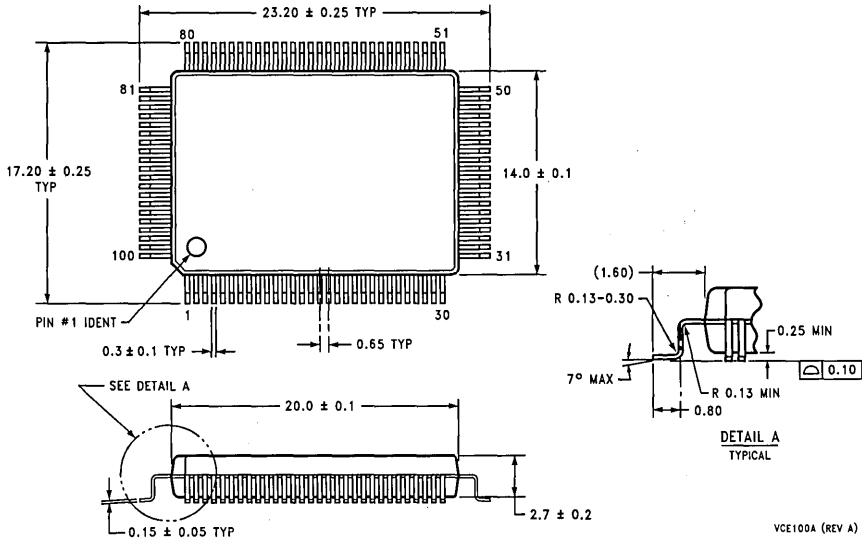
# 132 Lead Molded Plastic Leaded Chip Carrier NS Package Number V132A

All dimensions are in inches (millimeters)



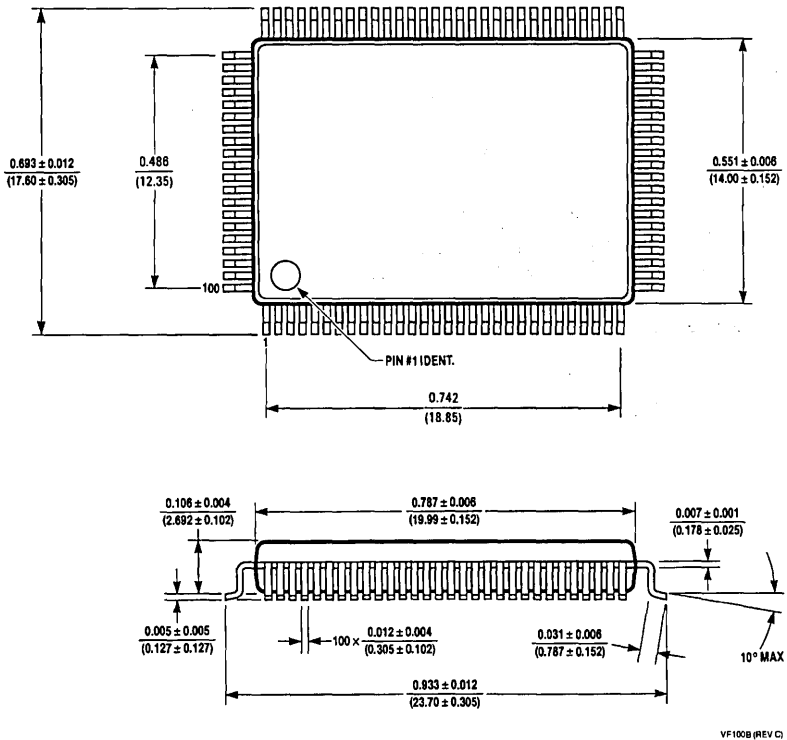
### 100 Lead (14mm x 20mm) Molded Plastic Quad Flat Package, JEDEC NS Package Number VCE100A

All dimensions are in millimeters



### 100 Lead Plastic Quad Flat Package, EIAJ NS Package Number VF100B

All dimensions are in inches (millimeters)

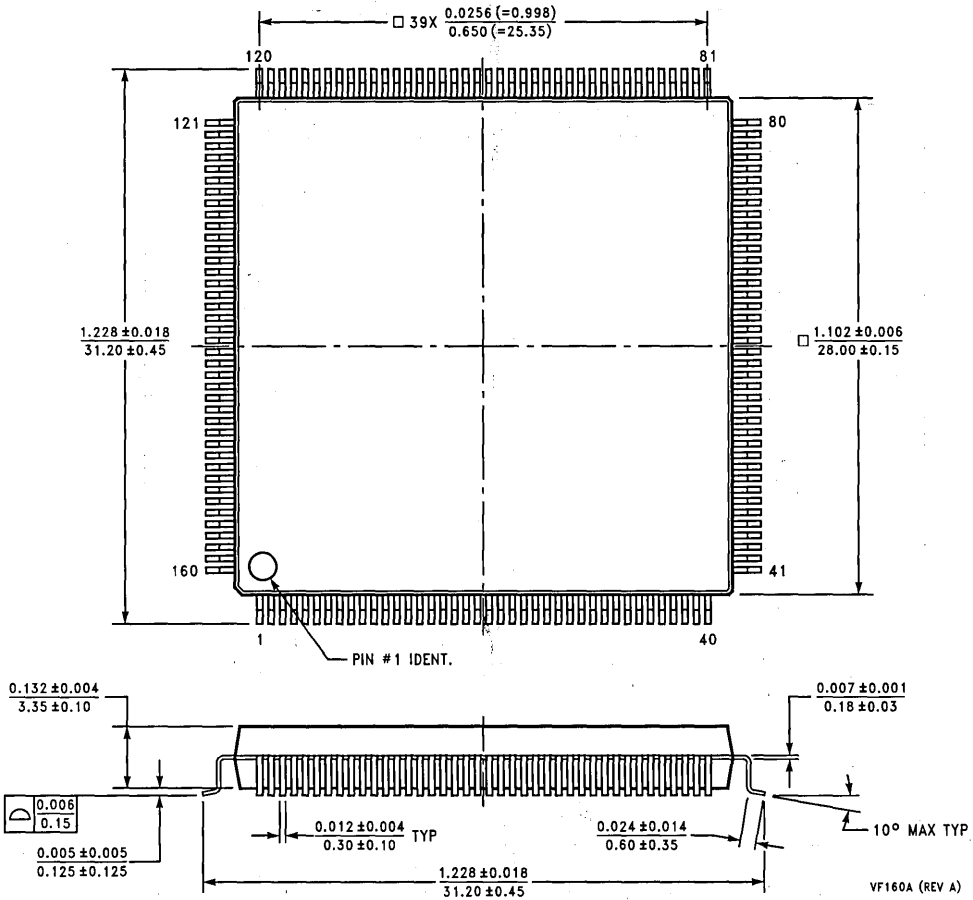






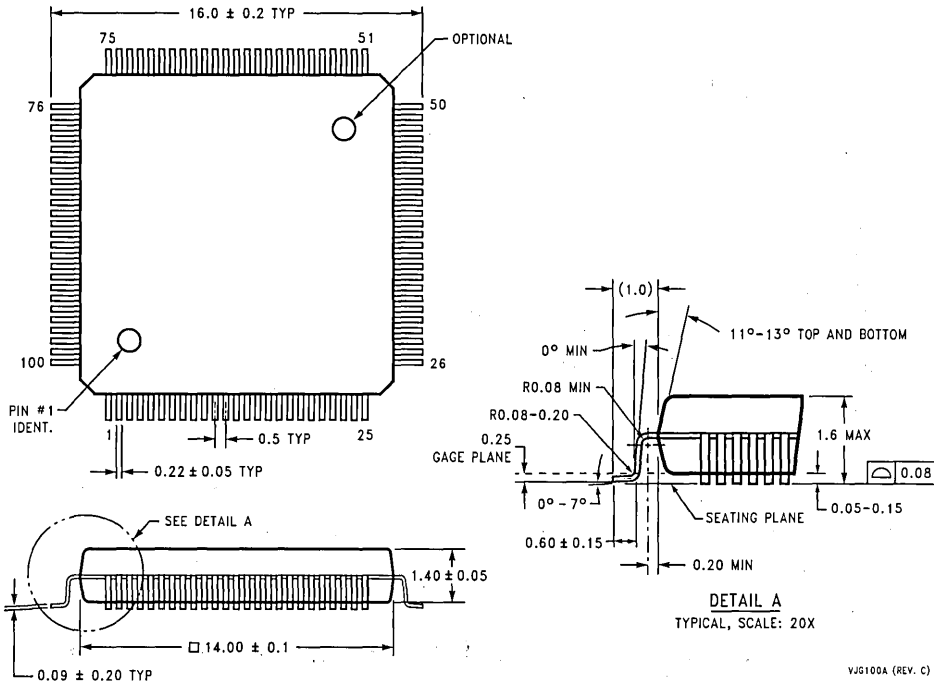
**160 Lead Plastic Quad Flat Package, EIAJ  
NS Package Number VF160A**

All dimensions are in  $\frac{\text{inches}}{\text{millimeters}}$



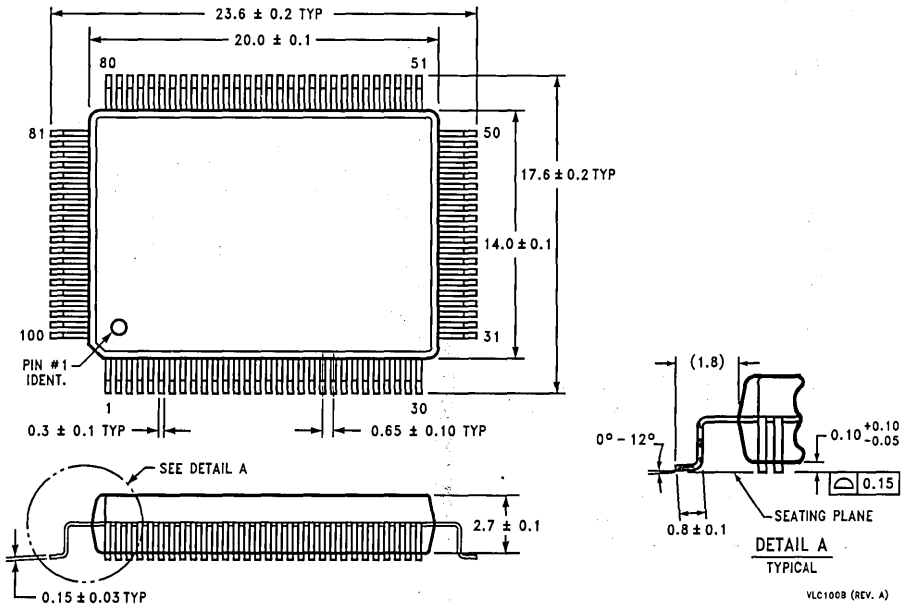
### 100 Lead (14mm x 14mm) Molded Plastic Quad Flat Package, JEDEC NS Package Number VJG100A

All dimensions are in millimeters



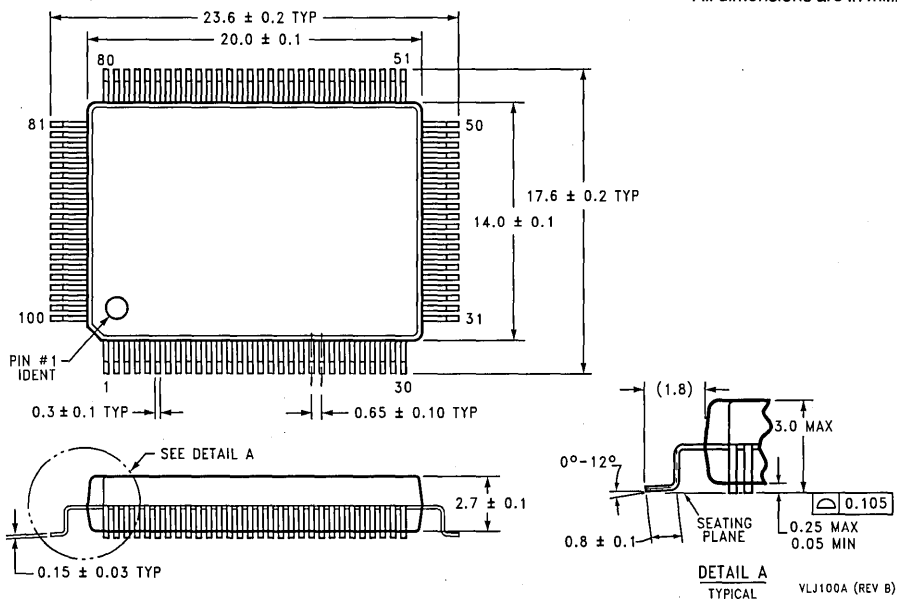
### 100 Lead JEDEC Metric Plastic Quad Flat Package NS Package Number VLC100B

All dimensions are in millimeters



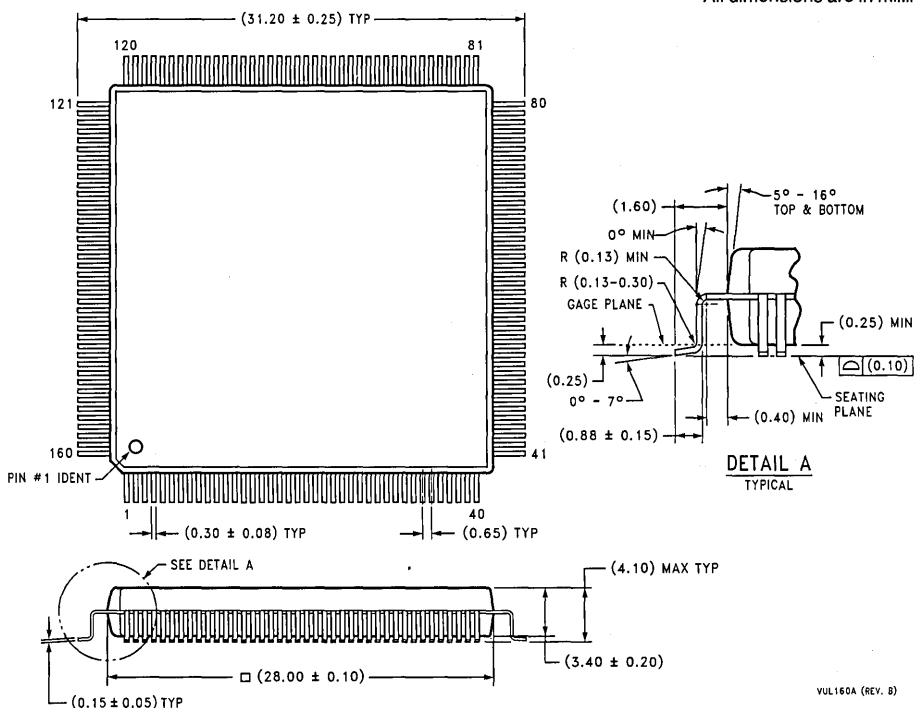
### 100 Lead (14mm x 20mm) Molded Plastic Quad Flat Package, EIAJ NS Package Number VLJ100A

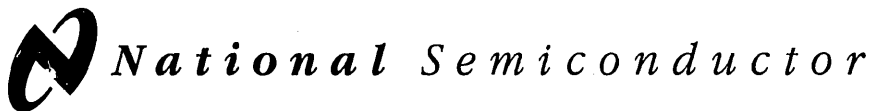
All dimensions are in millimeters



### 160 Lead (28mm x 28mm) Molded Plastic Quad Flat Package, JEDEC NS Package Number VUL160A

All dimensions are in millimeters





## **Bookshelf of Technical Support Information**

National Semiconductor Corporation recognizes the need to keep you informed about the availability of current technical literature.

This bookshelf is a compilation of books that are currently available. The listing that follows shows the publication year and section contents for each book.

For datasheets on new products and devices still in production but not found in a databook, please contact the National Semiconductor Customer Support Center at 1-800-272-9959.

We are interested in your comments on our technical literature and your suggestions for improvement.

Please send them to:

Technical Communications Dept. M/S 16-300  
2900 Semiconductor Drive  
P.O. Box 58090  
Santa Clara, CA 95052-8090

## **ADVANCED BiCMOS LOGIC (ABT, BiCMOS SCAN, LOW VOLTAGE BiCMOS, EXTENDED TTL TECHNOLOGY) DATABOOK—1996**

ABT Description and Family Characteristics • ABT Ratings, Specifications and Waveforms  
ABT Applications and Design Considerations • Quality and Reliability  
SCAN18xxxA BiCMOS 5V Logic with Boundary Scan • 74LVT Low Voltage BiCMOS Logic  
VME Extended TTL Technology for Backplanes • Advanced BiCMOS Clock Generation and Support

## **ADVANCED BIPOLAR LOGIC FAST, FASTr, ALS, AS DATABOOK—1995**

Introduction to Advanced Bipolar Logic Families • FAST/FASTr/ALS/AS • Family Characteristics  
Ratings, Specifications and Waveforms • Design Considerations • Datasheets • Ordering and Packaging Information

## **APPLICATION SPECIFIC ANALOG PRODUCTS DATABOOK—1995**

Audio Circuits • Video Circuits • Automotive • Special Functions • Surface Mount

## **ASIC DESIGN MANUAL/GATE ARRAYS & STANDARD CELLS—1987**

SSI/MSI Functions • Peripheral Functions • LSI/VLSI Functions • Design Guidelines • Packaging

## **CLOCK GENERATION AND SUPPORT (CGS) DESIGN DATABOOK—1995**

Low Skew Clock Buffers/Drivers • Video Clock Generators • Low Skew PLL Clock Generators  
Crystal Clock Oscillators

## **COP8™ DATABOOK—1994**

COP8 Family • COP8 Applications • MICROWIRE/PLUS Peripherals • COP8 Development Support

## **CROSSVOLT™ LOW VOLTAGE LOGIC SERIES DATABOOK—1996**

LCX Family • LVX Translator Family • LVX Bus Switch Family • LVX Family • LVQ Family • LVT Family  
ALCX Family • GTL Family

## **DATA ACQUISITION DATABOOK—1995**

Data Acquisition Systems • Analog-to-Digital Converters • Digital-to-Analog Converters • Voltage References  
Temperature Sensors • Active Filters • Analog Switches/Multiplexers • Surface Mount

## **DATA ACQUISITION DATABOOK SUPPLEMENT—1992**

New devices released since the printing of the 1989 Data Acquisition Linear Devices Databook.

## **DISCRETE SEMICONDUCTOR PRODUCTS DATABOOK—1989**

Selection Guide and Cross Reference Guides • Diodes • Bipolar NPN Transistors  
Bipolar PNP Transistors • JFET Transistors • Surface Mount Products • Pro-Electron Series  
Consumer Series • Power Components • Transistor Datasheets • Process Characteristics

## **DRAM MANAGEMENT HANDBOOK—1993**

Dynamic Memory Control • CPU Specific System Solutions • Error Detection and Correction  
Microprocessor Applications

## **EMBEDDED CONTROLLERS DATABOOK—1992**

COP400 Family • COP800 Family • COPS Applications • HPC Family • HPC Applications  
MICROWIRE and MICROWIRE/PLUS Peripherals • Microcontroller Development Tools

## **ETHERNET DATABOOK—1996**

Integrated Network Interface Controller Products • 10 Mb/s Physical Layer Transceivers and ENDECs  
10 Mb/s Repeater Interface Controller Products • 100 Mb/s Fast Ethernet Protocol Products  
Glossary and Acronyms

## **FDDI DATABOOK—1994**

Datasheets • Application Notes

## **F100K ECL LOGIC DATABOOK & DESIGN GUIDE—1992**

Family Overview • 300 Series (Low-Power) Datasheets • 100 Series Datasheets • 11C Datasheets  
Design Guide • Circuit Basics • Logic Design • Transmission Line Concepts • System Considerations  
Power Distribution and Thermal Considerations • Testing Techniques • 300 Series Package Qualification  
Quality Assurance and Reliability • Application Notes

## **FACT™ ADVANCED CMOS LOGIC DATABOOK—1993**

Description and Family Characteristics • Ratings, Specifications and Waveforms  
Design Considerations • 54AC/74ACXXX • 54ACT/74ACTXXX • Quiet Series: 54ACQ/74ACQXXX  
Quiet Series: 54ACTQ/74ACTQXXX • 54FCT/74FCTXXX • FCTA: 54FCTXXXA/74FCTXXXA/B

## **FAST® APPLICATIONS HANDBOOK—1990**

Reprint of 1987 Fairchild FAST Applications Handbook

Contains application information on the FAST family: Introduction • Multiplexers • Decoders • Encoders  
Operators • FIFOs • Counters • TTL Small Scale Integration • Line Driving and System Design  
FAST Characteristics and Testing • Packaging Characteristics

## **HIGH-PERFORMANCE BUS INTERFACE DATABOOK—1994**

QuickRing • Futurebus+ /BTL Devices • BTL Transceiver Application Notes • Futurebus+ Application Notes  
High Performance TTL Bus Drivers • PI-Bus • Futurebus+ /BTL Reference

## **IBM DATA COMMUNICATIONS HANDBOOK—1992**

IBM Data Communications • Application Notes

## **INTERFACE: DATA TRANSMISSION DATABOOK—1994**

TIA/EIA-232 (RS-232) • TIA/EIA-422/423 • TIA/EIA-485 • Line Drivers • Receivers • Repeaters  
Transceivers • Low Voltage Differential Signaling • Special Interface • Application Notes

## **LINEAR APPLICATIONS HANDBOOK—1994**

The purpose of this handbook is to provide a fully indexed and cross-referenced collection of linear integrated circuit applications using both monolithic and hybrid circuits from National Semiconductor.

Individual application notes are normally written to explain the operation and use of one particular device or to detail various methods of accomplishing a given function. The organization of this handbook takes advantage of this innate coherence by keeping each application note intact, arranging them in numerical order, and providing a detailed Subject Index.

## **LOW VOLTAGE DATABOOK—1992**

This databook contains information on National's expanding portfolio of low and extended voltage products. Product datasheets included for: Low Voltage Logic (LVQ), Linear, EPROM, EEPROM, SRAM, Interface, ASIC, Embedded Controllers, Real Time Clocks, and Clock Generation and Support (CGS).

## **MASS STORAGE HANDBOOK—1989**

Rigid Disk Pulse Detectors • Rigid Disk Data Separators/Synchronizers and ENDECs  
Rigid Disk Data Controller • SCSI Bus Interface Circuits • Floppy Disk Controllers • Disk Drive Interface Circuits  
Rigid Disk Preamplifiers and Servo Control Circuits • Rigid Disk Microcontroller Circuits • Disk Interface Design Guide

## **MEMORY DATABOOK—1994**

FLASH • CMOS EPROMs • CMOS EEPROMs • PROMs • Application Notes

## **MEMORY APPLICATIONS HANDBOOK—1994**

FLASH • EEPROMs • EPROMs • Application Notes

## **OPERATIONAL AMPLIFIERS DATABOOK—1995**

Operational Amplifiers • Buffers • Voltage Comparators • Active Matrix/LCD Display Drivers  
Special Functions • Surface Mount

## **PACKAGING DATABOOK—1993**

Introduction to Packaging • Hermetic Packages • Plastic Packages • Advanced Packaging Technology  
Package Reliability Considerations • Packing Considerations • Surface Mount Considerations

## **POWER IC's DATABOOK—1995**

Linear Voltage Regulators • Low Dropout Voltage Regulators • Switching Voltage Regulators  
Motion Control • Surface Mount

## **PRODUCTS FOR WIRELESS COMMUNICATIONS—1996**

Radio Transceiver Components • Baseband Processing Components • Control and Signal Processing Components  
Non-Volatile Memory • Audio Interface Components • Support Circuitry • Power Management  
Complete Cordless Phone Solution

## **PROGRAMMABLE LOGIC DEVICE DATABOOK AND DESIGN GUIDE—1993**

Product Line Overview • Datasheets • Design Guide: Designing with PLDs • PLD Design Methodology  
PLD Design Development Tools • Fabrication of Programmable Logic • Application Examples

## **REAL TIME CLOCK HANDBOOK—1993**

3-Volt Low Voltage Real Time Clocks • Real Time Clocks and Timer Clock Peripherals • Application Notes

## **RELIABILITY HANDBOOK—1987**

Reliability and the Die • Internal Construction • Finished Package • MIL-STD-883 • MIL-M-38510  
The Specification Development Process • Reliability and the Hybrid Device • VLSI/VHSIC Devices  
Radiation Environment • Electrostatic Discharge • Discrete Device • Standardization  
Quality Assurance and Reliability Engineering • Reliability and Documentation • Commercial Grade Device  
European Reliability Programs • Reliability and the Cost of Semiconductor Ownership  
Reliability Testing at National Semiconductor • The Total Military/Aerospace Standardization Program  
883B/RETS™ Products • MILS/RETS™ Products • 883/RETS™ Hybrids • MIL-M-38510 Class B Products  
Radiation Hardened Technology • Wafer Fabrication • Semiconductor Assembly and Packaging  
Semiconductor Packages • Glossary of Terms • Key Government Agencies • AN/ Numbers and Acronyms  
Bibliography • MIL-M-38510 and DESC Drawing Cross Listing

## **SCAN™ DATABOOK—1994**

Evolution of IEEE 1149.1 Standard • SCAN BiCMOS Products • SCAN CMOS Products • System Test Products  
Other IEEE 1149.1 Devices

## **TELECOMMUNICATIONS—1994**

COMBO and SLIC Devices • ISDN • Digital Loop Devices • Analog Telephone Components • Software • Application Notes

## **VHC ADVANCED CMOS LOGIC DATABOOK—1996**

This databook introduces National's Very High Speed CMOS (VHC) and Very High Speed TTL Compatible CMOS (VHCT) designs. The databook includes Description and Family Characteristics • Ratings, Specifications and Waveforms  
Design Considerations • VHC Family Datasheets • VHC Specialty Function Datasheets and related Application Notes.  
The topics discussed are the advantages of VHC/VHCT AC Performance, Low Noise Characteristics and Improved Interface Capabilities.

# NATIONAL SEMICONDUCTOR CORPORATION DISTRIBUTORS

## ALABAMA

Huntsville  
Anthem Electronics  
(205) 890-0302  
Future Electronics Corp.  
(205) 830-2322  
Hamilton/Hallmark  
(205) 837-8700  
Pioneer Technology  
(205) 837-9300  
Time Electronics  
(205) 721-1134

## ARIZONA

Phoenix  
Future Electronics Corp.  
(602) 968-7140  
Hamilton/Hallmark  
(602) 437-1200  
Scottsdale  
Alliance Electronics Inc.  
(602) 483-9400  
Tempe  
Anthem Electronics  
(602) 966-6600  
Bell Industries  
(602) 966-3600  
Pioneer Standard  
(602) 350-9335  
Time Electronics  
(602) 967-2000

## CALIFORNIA

Agoura Hills  
Future Electronics Corp.  
(818) 865-0040  
Pioneer Standard  
(818) 865-5800  
Time Electronics  
(818) 707-2890  
Chatsworth  
Anthem Electronics  
(818) 775-1333  
Costa Mesa  
Hamilton/Hallmark  
(714) 641-4100  
Irvine  
Anthem Electronics  
(714) 768-4444  
Bell Industries  
(714) 727-4500  
Future Electronics Corp.  
(714) 453-1515  
Pioneer Standard  
(714) 753-5090  
Zeus Elect. an Arrow Co.  
(714) 581-4622  
Rocklin  
Anthem Electronics  
(916) 624-9744  
Bell Industries  
(916) 652-0418  
Roseville  
Future Electronics Corp.  
(916) 783-7877  
Hamilton/Hallmark  
(916) 624-9781

San Diego  
Anthem Electronics  
(619) 453-9005  
Bell Industries  
(619) 576-3294  
Future Electronics Corp.  
(619) 625-2800  
Hamilton/Hallmark  
(619) 571-7540  
Pioneer Standard  
(619) 514-7700  
Time Electronics  
(619) 674-2800  
San Jose  
Anthem Electronics  
(408) 453-1200  
Future Electronics Corp.  
(408) 434-1122

San Jose (Continued)  
Hamilton/Hallmark  
(408) 435-3500  
Pioneer Technology  
(408) 954-9100  
Zeus Elect. an Arrow Co.  
(408) 629-4789

Sunnyvale  
Bell Industries  
(408) 734-8570  
Time Electronics  
(408) 734-9890

Tustin  
Time Electronics  
(714) 669-0216

Westlake Village  
Bell Industries  
(805) 373-5600

Woodland Hills  
Hamilton/Hallmark  
(818) 594-0404  
Time Electronics  
(818) 593-8400

## COLORADO

Denver  
Bell Industries  
(303) 691-9270  
Englewood  
Anthem Electronics  
(303) 790-4500  
Hamilton/Hallmark  
(303) 790-1662  
Pioneer Technology  
(303) 773-8090  
Time Electronics  
(303) 799-5400  
Lakewood  
Future Electronics Corp.  
(303) 292-2008

## CONNECTICUT

Cheshire  
Future Electronics Corp.  
(203) 250-0083  
Hamilton/Hallmark  
(203) 271-2844  
Meriden  
Bell Industries  
(203) 639-6000  
Shelton  
Pioneer Standard  
(203) 929-5600  
Wallingford  
Advent Electronics  
(800) 982-0014  
Waterbury  
Anthem Electronics  
(203) 575-1575

## FLORIDA

Altamonte Springs  
Anthem Electronics  
(407) 831-0007  
Bell Industries  
(407) 339-0078  
Future Electronics Corp.  
(407) 865-7900  
Pioneer Technology  
(407) 834-9090  
Deerfield Beach  
Future Electronics Corp.  
(305) 428-4043  
Pioneer Technology  
(305) 428-8877  
Fort Lauderdale  
Hamilton/Hallmark  
(305) 484-5482  
Time Electronics  
(305) 484-1864  
Indianantic  
Advent Electronics  
(800) 975-8669  
Lake Mary  
Zeus Elect. an Arrow Co.  
(407) 333-9300

## Largo

Future Electronics Corp.  
(813) 530-1222  
Hamilton/Hallmark  
(813) 541-7440

## Orlando

Chip Supply  
"Die Distributor"  
(407) 298-7100  
Time Electronics  
(407) 841-6566

## Winter Park

Hamilton/Hallmark  
(407) 657-3300

## GEORGIA

Duluth  
Anthem Electronics  
(404) 931-9300  
Hamilton/Hallmark  
(404) 623-4400  
Pioneer Technology  
(404) 623-1003  
Time Electronics  
(404) 623-5455  
Norcross  
Future Electronics Corp.  
(404) 441-7676

## ILLINOIS

Addison  
Pioneer Standard  
(708) 495-9680  
Bensenville  
Hamilton/Hallmark  
(708) 860-7780  
Des Plaines  
Advent Electronics  
(800) 323-1270  
Elk Grove Village  
Bell Industries  
(708) 640-1910  
Hoffman Estates  
Future Electronics Corp.  
(708) 882-1255  
Itasca  
Zeus Elect. an Arrow Co.  
(708) 595-9730  
Schaumburg  
Anthem Electronics  
(708) 884-0200  
Time Electronics  
(708) 303-3000

## INDIANA

Carmel  
Hamilton/Hallmark  
(317) 575-3500  
Fort Wayne  
Bell Industries  
(219) 422-4300  
Indianapolis  
Advent Electronics Inc.  
(800) 732-1453  
Bell Industries  
(317) 875-8200  
Future Electronics Corp.  
(317) 469-0447  
Pioneer Standard  
(317) 573-0880

## IOWA

Cedar Rapids  
Advent Electronics  
(800) 397-8407  
Hamilton/Hallmark  
(319) 393-0033

## KANSAS

Lenexa  
Hamilton/Hallmark  
(913) 888-4747  
Overland Park  
Future Electronics Corp.  
(913) 649-1531

## KENTUCKY

Lexington  
Hamilton/Hallmark  
(606) 288-4911

## MARYLAND

Columbia  
Anthem Electronics  
(410) 995-6640  
Bell Industries  
(410) 290-5100  
Future Electronics Corp.  
(410) 290-0600  
Hamilton/Hallmark  
(410) 988-9800  
Seymour Electronics  
(410) 992-7474  
Time Electronics  
(410) 720-3600  
Gaithersburg  
Pioneer Technology  
(301) 921-0660

## MASSACHUSETTS

Andover  
Bell Industries  
(508) 474-8880  
Bolton  
Future Electronics Corp.  
(508) 779-3000  
Lexington  
Pioneer Standard  
(617) 861-9200  
Newburyport  
Rochester Electronics  
"Obsolete Products"  
(508) 462-9332  
Norwood  
Gerber Electronics  
(617) 769-6000  
Peabody  
Hamilton/Hallmark  
(508) 532-3701  
Time Electronics  
(508) 532-9777  
Wilmington  
Anthem Electronics  
(508) 657-5170  
Zeus Elect. an Arrow Co.  
(508) 658-0900

## MICHIGAN

Farmington Hills  
Advent Electronics  
(800) 572-9329  
Grand Rapids  
Future Electronics Corp.  
(616) 698-6800  
Pioneer Standard  
(616) 698-1800  
Livonia  
Future Electronics Corp.  
(313) 261-5270  
O'Fallon  
Advent Electronics  
(800) 888-9588  
Plymouth  
Hamilton/Hallmark  
(313) 416-5800  
Pioneer Standard  
(313) 416-2157

## MINNESOTA

Bloomington  
Hamilton/Hallmark  
(612) 881-2600  
Eden Prairie  
Anthem Electronics  
(612) 944-5454  
Future Electronics Corp.  
(612) 944-2200  
Pioneer Standard  
(612) 944-3355  
Minnetonka  
Time Electronics  
(612) 931-2131  
Thief River Falls  
Digi-Key Corp.  
"Catalog Sales Only"  
(800) 344-4539



# NATIONAL SEMICONDUCTOR CORPORATION DISTRIBUTORS (Continued)

## MISSOURI

Eart., City  
Hamilton/Hallmark  
(314) 291-5350  
Manchester  
Time Electronics  
(314) 230-7500  
St. Louis  
Future Electronics Corp.  
(314) 469-6805

## NEW JERSEY

Camden  
Advent Electronics  
(800) 255-4771  
Cherry Hill  
Hamilton/Hallmark  
(609) 424-0110  
Fairfield  
Bell Industries  
(201) 227-6060  
Pioneer Standard  
(201) 575-3510  
Marlton  
Future Electronics Corp.  
(609) 596-4080  
Time Electronics  
(609) 596-1286  
Mount Laurel  
Bell Industries  
(609) 439-8860  
Seymour Electronics  
(609) 235-7474  
Parsippany  
Future Electronics Corp.  
(201) 299-0400  
Hamilton/Hallmark  
(201) 515-1641  
Pine Brook  
Anthem Electronics  
(201) 227-7960  
Wayne  
Time Electronics  
(201) 785-8250

## NEW MEXICO

Albuquerque  
Bell Industries  
(505) 292-2700  
Hamilton/Hallmark  
(505) 828-1058

## NEW YORK

Binghamton  
Pioneer Standard  
(607) 722-9300  
Commack  
Anthem Electronics  
(516) 864-6600  
Fairport  
Pioneer Standard  
(716) 381-7070  
Hauppauge  
Future Electronics Corp.  
(516) 234-4000  
Hamilton/Hallmark  
(516) 434-7400  
Time Electronics  
(516) 273-0100  
Port Chester  
Zeus Elect. an Arrow Co.  
(914) 937-7400  
Rochester  
Future Electronics Corp.  
(716) 387-9550  
Hamilton/Hallmark  
(800) 475-9130  
Syracuse  
Future Electronics Corp.  
(315) 451-2371  
Time Electronics  
(315) 434-9837

Woodbury  
Pioneer Standard  
(516) 921-9700  
Seymour Electronics  
(516) 496-7474

## NORTH CAROLINA

Charlotte  
Future Electronics Corp.  
(704) 547-1107  
Morrisville  
Pioneer Technology  
(919) 460-1530  
Raleigh  
Anthem Electronics  
(919) 782-3550  
Future Electronics Corp.  
(919) 790-7111  
Hamilton/Hallmark  
(919) 872-0712

## OHIO

Beavercreek  
Future Electronics Corp.  
(513) 426-0090  
Cleveland  
Pioneer Standard  
(216) 587-3600  
Columbus  
Time Electronics  
(614) 794-3301  
Dayton  
Bell Industries  
(513) 435-5922  
Bell Industries-Military  
(513) 434-8231  
Hamilton/Hallmark  
(513) 439-6735  
Pioneer Standard  
(513) 236-9900  
Mayfield Heights  
Future Electronics Corp.  
(216) 449-6996  
Solon  
Bell Industries  
(216) 498-2002  
Hamilton/Hallmark  
(216) 498-1100  
Worthington  
Hamilton/Hallmark  
(614) 888-3313

## OKLAHOMA

Tulsa  
Hamilton/Hallmark  
(918) 254-6110  
Pioneer Standard  
(918) 665-7840  
Radio Inc.  
(918) 587-9123

## OREGON

Beaverton  
Anthem Electronics  
(503) 643-1114  
Bell Industries  
(503) 644-3444  
Future Electronics Corp.  
(503) 645-9454  
Hamilton/Hallmark  
(503) 526-6200  
Pioneer Technology  
(503) 626-7300  
Portland  
Time Electronics  
(503) 684-3780

## PENNSYLVANIA

Horsham  
Anthem Electronics  
(215) 443-5150  
Pioneer Technology  
(215) 674-4000  
Pittsburgh  
Pioneer Standard  
(412) 782-2300

## TEXAS

Austin  
Anthem Electronics  
(512) 388-0049  
Future Electronics Corp.  
(512) 502-0991  
Hamilton/Hallmark  
(512) 258-8848  
Minco Technology Labs.  
"Die Distributor"  
(512) 834-2022  
Pioneer Standard  
(512) 835-4000  
Time Electronics  
(512) 219-3773  
Carrollton  
Zeus Elect. an Arrow Co.  
(214) 380-4330  
Dallas  
Hamilton/Hallmark  
(214) 553-4300  
Pioneer Standard  
(214) 386-7300  
Houston  
Future Electronics Corp.  
(713) 785-1155  
Hamilton/Hallmark  
(713) 781-6100  
Pioneer Standard  
(713) 495-4700  
Richardson  
Anthem Electronics  
(214) 238-7100  
Bell Industries  
(214) 690-9096  
Future Electronics Corp.  
(214) 437-2437  
Time Electronics  
(214) 480-5000

## UTAH

Midvale  
Bell Industries  
(801) 255-9691  
Salt Lake City  
Anthem Electronics  
(801) 973-8555  
Future Electronics Corp.  
(801) 467-4448  
Hamilton/Hallmark  
(801) 266-2022  
West Valley City  
Time Electronics  
(801) 973-0208

## WASHINGTON

Bellevue  
Bell Industries  
(206) 646-8750  
Pioneer Technology  
(206) 644-7500  
Bothell  
Anthem Electronics  
(206) 483-1700  
Future Electronics Corp.  
(206) 489-3400  
Kirkland  
Time Electronics  
(206) 820-1525  
Redmond  
Hamilton/Hallmark  
(206) 881-6697

## WISCONSIN

Brookfield  
Future Electronics Corp.  
(414) 879-0244  
Pioneer Standard  
(414) 784-3480  
Mequon  
Taylor Electric  
(414) 241-4321

New Berlin  
Hamilton/Hallmark  
(414) 780-7200  
Waukesha  
Bell Industries  
(414) 547-8879  
West Allis  
Advent Electronics  
(800) 500-0441

## CANADA

### WESTERN PROVINCES

Burnaby  
Hamilton/Hallmark  
(604) 420-4101  
Calgary  
Electro Sonic Inc.  
(403) 255-9550  
Future Electronics Corp.  
(403) 250-5550  
Zentronics/Pioneer  
(403) 295-8838  
Edmonton  
Future Electronics Corp.  
(403) 438-2858  
Zentronics/Pioneer  
(403) 482-3038  
Richmond  
Electro Sonic Inc.  
(604) 273-2911  
Zentronics/Pioneer  
(604) 273-5575  
Vancouver  
Future Electronics Corp.  
(604) 294-1166

### EASTERN PROVINCES

Mississauga  
Future Electronics Corp.  
(905) 612-9200  
Hamilton/Hallmark  
(905) 564-6060  
Time Electronics  
(905) 712-3277  
Zentronics/Pioneer  
(905) 405-8300  
Nepean  
Hamilton/Hallmark  
(613) 226-1700  
Zentronics/Pioneer  
(613) 226-8840  
Ottawa  
Electro Sonic Inc.  
(613) 728-8333  
Future Electronics Corp.  
(613) 820-8313  
Pointe Claire  
Future Electronics Corp.  
(514) 694-7710  
Quebec  
Future Electronics Corp.  
(418) 877-6666  
Ville St. Laurent  
Hamilton/Hallmark  
(514) 335-1000  
Zentronics/Pioneer  
(514) 737-9700  
Willowdale  
Electro Sonic Inc.  
(416) 694-1666  
Winnipeg  
Electro Sonic Inc.  
(204) 783-3105  
Future Electronics Corp.  
(204) 944-1446  
Zentronics/Pioneer  
(204) 694-1957

**WORLDWIDE SALES OFFICES**
**AUSTRALIA**

**National Semiconductor (Australia) Pty. Ltd.**  
Bldg. 16 Business Park Dr.  
Monash Business Park  
Nottingham Melbourne  
Victoria 3168 Australia  
Tel: (39) 558-9999  
Fax: (39) 558-9998

**BRAZIL**

**National Semiconductores Do Brazil Ltda.**  
Rue Deputado Lacerda  
Franco 120-3A  
Sao Paulo-SP Brazil 05418-000  
Tel: (55-11) 212-5066  
Fax: (55-11) 212-1181

**CANADA**

**National Semiconductor (Canada)**  
5925 Airport Road, Suite 615  
Mississauga, Ontario L4V 1W1  
Tel: (416) 678-2920  
Fax: (416) 678-2837

**National Semiconductor (Canada)**  
39 Robertson Road, Suite 101  
Nepean, Ontario K2H 8R2  
Tel: (613) 596-0411  
Fax: (613) 596-1613

**National Semiconductor (Canada)**  
1870 Boul Des Sources,  
Suite 101  
Pointe Claire, Quebec H2R 5N4  
Tel: (514) 426-2992  
Fax: (514) 426-2710

**CHINA**

**National Semiconductor Beijing China Liaison Office**  
Room 613 & 614  
Sinochem Mansion  
No. A2 Fuxingmenwai Avenue  
Beijing 100046, PRC  
China  
Tel: 86-10-8568601  
Fax: 86-10-8568606

**National Semiconductor Shanghai China Liaison Office**  
B702, Universal Mansion  
No. 172, Yuyuan Road  
Shanghai 200040, PRC  
China  
Tel: 86-21-2496062  
Fax: 86-21-2496063

**FINLAND**

**National Semiconductor (U.K.) Ltd.**  
Mekaanikonkatu 13  
SF-00810 Helsinki  
Finland  
Tel: (0) 759-1855  
Fax: (0) 759-1393

**FRANCE**

**National Semiconductor S.A.**  
Parc d'Affaires Technopolis  
3, Avenue Du Canada  
Bat. ZETA - L.P. 821 Les Ulis  
F-91974 Courtabouef Cedex  
France  
Tel: (1) 69 18 37 00  
Fax: (1) 69 18 37 69

**GERMANY**

**National Semiconductor GmbH**  
Livry-Gargan-Strasse. 10  
D-82256 Fürstenfeldbruck  
Germany  
Tel: (0-81-41) 35-0  
Fax: (0-81-41) 35-15-06

**HONG KONG**

**National Semiconductor Hong Kong Ltd.**  
13th Floor, Straight Block  
Ocean Centre  
5 Canton Road  
Tsimshatsui, Kowloon  
Hong Kong  
Tel: (852) 2737-1600  
Fax: (852) 2736-9960

**INDIA**

**National Semiconductor India Liaison Office**  
1109, 11th Floor, West Wing  
Raheja Towers, M.G. Road  
Bangalore 560001 India  
Tel: 91-80-559-9467  
Fax: 91-80-559-9468

**ISRAEL**

**National Semiconductor Ltd.**  
Mäskit Street  
PO Box 3007  
Herzlia B. 46104  
Israel  
Tel: (09) 59 42 55  
Fax: (09) 55 83 22

**ITALY**

**National Semiconductor S.p.A.**  
Strada 7, Palazzo R/3  
I-20089 Rozzano-Milano Fiori  
Italy  
Tel: (02) 57 50 03 00  
Fax: (02) 57 50 04 00

**JAPAN**

**National Semiconductor Japan Ltd.**  
Sumitomo Chemical  
Engineering Center Bldg. 7F  
1-7-1, Nakase, Mihama-Ku  
Chiba-City,  
Chiba Prefecture 261  
Japan  
Tel: (043) 299-2300  
Fax: (043) 299-2500

**KOREA**

**National Semiconductor (Far East) Ltd.**  
13th Floor, Dai Han  
Life Insurance 63 Building  
60 Yoido-Dong  
Youngdeungpo-KU  
Seoul Korea 150-763  
Tel: (02) 784-8051/3  
(02) 785-0696/8  
Fax: (02) 784-8054

**MALAYSIA**

**National Semiconductor Sdn Bhd**  
Bayan Lepas Free Trade Zone  
11900 Penang Malaysia  
Tel: 4-644-9061  
Fax: 4-644-9073

**MEXICO**

**Electronica NSC de Mexico SA**  
Juventino Rosas No. 118-2  
Col Guadalupe Inn  
Mexico, 01020 D.E. Mexico  
Tel: (525) 661-7155  
Fax: (525) 661-6905

**PUERTO RICO**

**National Semiconductor (Puerto Rico)**  
La Electronica Bldg.  
Suite 312, R.D. #1 KM 14.5  
Rio Piedras  
Puerto Rico 00927  
Tel: (809) 758-9211  
Fax: (809) 763-6959

**SINGAPORE**

**National Semiconductor Asia Pacific Pte. Ltd.**  
200 Cantonment Road # 13-01  
Southpoint Singapore 0208  
Tel: (65) 225-2226  
Fax: (65) 225-7080

**SPAIN**

**National Semiconductor GmbH**  
Calle Agustín de Foxa, 27 (9ºD)  
E-28036 Madrid  
Spain  
Tel: (01) 7-33-29-54  
Fax: (01) 7-33-80-18

**SWEDEN**

**National Semiconductor AB**  
P.O. Box 1009  
Grosshandlarvägen 7  
S-12123 Johanneshov,  
Sweden  
Tel: (08) 7 22 80 50  
Fax: (08) 7 22 90 95

**SWITZERLAND**

**National Semiconductor (U.K.) Ltd.**  
Alte Winterthurerstrasse 53  
CH-8304 Wallisellen-Zürich  
Switzerland  
Tel: (01) 8-30-27-27  
Fax: (01) 8-30-19-00

**TAIWAN**

**National Semiconductor (Far East) Ltd.**  
9/F, No. 44 Section 2  
Chungshan North Road  
Taipei, Taiwan, R.O.C.  
Tel: (02) 521-3288  
Fax: (02) 561-3054

**U.K. AND IRELAND**

**National Semiconductor (U.K.) Ltd.**  
1st Floor  
Milford House  
Milford Street  
Swindon, Wiltshire SN1 1DW  
United Kingdom  
Tel: (07-93) 61 41 41  
Fax: (07-93) 52 21 80  
Telex: 444674

**UNITED STATES**

**National Semiconductor Corporation**  
1111 West Bardin Road  
Arlington, TX 76017  
Tel: (800) 272-9959  
Fax: (800) 737-7018

*National Semiconductor supplies a comprehensive set of service and support capabilities. Complete product information and design support is available from National's customer support centers.*

*To receive sales literature and technical assistance, contact the National support center in your area.*



**Americas** Tel: 1-800-272-9959  
Fax: 1-800-737-7018  
Email: support@tevm2.nsc.com

**Europe** Fax: (+49) 0-180-530 85 86  
Email: cnjwge@tevm2.nsc.com

Deutsch Tel: (+49) 0-180-530 85 85  
English Tel: (+49) 0-180-532 78 32  
Français Tél: (+49) 0-180-532 93 58  
Italiano Tel: (+49) 0-180-534 16 80

**Japan** Tel: 81-043-299-2309  
Fax: 81-043-299-2408

*See us on the Worldwide Web at: <http://www.nsc.com>*

For support in the following countries, please contact the offices listed below:

**Australia**

Tel: (39) 558-9999  
Fax: (39) 558-9998

**China**

Tel: 10-849-133 1  
Fax: 10-849-133 2

**Hong Kong**

Tel: (852) 2737-1600  
Fax: (852) 2736-9960

**India**

Tel: 80-226-7272  
Fax: 80-225-1133

**Korea**

Tel: (02) 784-8051/3  
(02) 785-0696/8  
Fax: (02) 784-8054

**Malaysia**

Tel: 4-644-9061  
Fax: 4-644-9073

**Singapore**

Tel: (65) 225-2226  
Fax: (65) 225-7080

**Taiwan**

Tel: (02) 521-3288  
Fax: (02) 561-3054

For a complete listing of worldwide sales offices, see inside back page.